NASA Reference Publication 1338

1994

NASA Geometry Data Exchange Specification for Computational Fluid Dynamics (NASA IGES)

Matthew W. Blake, *Ames Research Center, Moffett Field, California*
Patricia A. Kerr, *Langley Research Center, Hampton, Virginia*
Scott A. Thorp, *Lewis Research Center, Cleveland, Ohio*
Jin J. Chou, *Computer Sience Corporation, Ames Research Center, Moffett Field, California*

National Aeronautics and Space Administration

Ames Research Center
Moffett Field, California 94035-1000

TABLE OF CONTENTS

1.0 Introduction

1.1 Background

The geometry data received by NASA scientists for analysis and modification is currently supplied in numerous formats which often require hundreds of hours of manipulation to achieve a format capable of being utilized by analysis software. This modified data set usually has lost a level of accuracy from the original data and often may not maintain the design intent of the original data as developed on the original designer's system.

In the spring of 1991, the NASA Surface Modeling and Grid Generation Steering Committee determined that one of the leading detriments to the grid generation process was the lack of a standard method of transferring complex vehicle geometries between various software systems. A subcommittee of technical personnel from the Ames, Langley, and Lewis Research Centers was formed to develop a data exchange format.

Following an analysis of existing and proposed standards, the subcommittee selected the existing Initial Graphics Exchange Specification (IGES) format (ref. 1) as the basis for a NASA standard. In the United States, IGES is by far the most widely used product data exchange specification. The latest version of the IGES specification (version 5.1) provides an adequate set of geometric entities to cover the current data transfer needs for computational fluid dynamics (CFD) research.

A subset of the IGES capability was selected, and a draft NASA Technical Specification was released in September of 1991 titled "NASA Geometry Data Exchange Specification Utilizing IGES." In the specification, the Rational B-spline was chosen as the most stable format to represent all types of geometry and was selected as the primary geometry representation method. In April of 1992, this subset of entities was proposed to the IGES/PDES Organization (IPO) for acceptance as an official IGES Application Protocol (AP). The IPO did not feel comfortable with restricting geometry entities to a limited subset in an AP. As the restriction on entities was the key to the useability of this specification, the NASA geometry subcommittee chose to proceed with the completion of this document and the development of software to utilize data based on this standard without pursuing official IPO acceptance.

Since files conforming to this specification are valid IGES files, there should be minimal impact on industry conversion to utilizing NASA-IGES.

The standard IGES file format is very complex. The IGES documentation is also very large and complex. Utilizing IGES data files requires expert knowledge of the format. Even though this specification contains significantly fewer entities, it still inherits a major portion of the complexity of the IGES file format. It is unreasonable to expect most scientists and CFD software developers to spend the time necessary to understand the file format and to handle the files directly. This IGES file complexity problem has lead to the development of the main body of this specification.

Please note that the IGES entities allowed under this specification and other related information are contained in Appendix A of this document. Reference in this document to "NASA-IGES specification" or "NASA-IGES files" refers to the subset of IGES entities specified in Appendix A and IGES files conforming to that specification.

1.2 Purpose

This specification is intended to provide a definition of the geometry of an object used in CFD analysis and a method for exchanging that information. This specification should facilitate the usage of the NASA-IGES protocol for rapid and accurate data transfer, and to promote the use of an accurate and unified geometry representation method for CFD research. These goals are achieved through an easy-to-understand mathematical description of the geometry and an IGES-independent data description in the main body of this document. Appendix A contains the "hooks" into the IGES-specific method of implementing this geometry data description.

It is hoped that this document will help software designers develop analysis codes that work directly on the CAD data and that more accurate analysis can be performed by utilizing accurate geometry data.

1.3 Scope

This document describes in detail the geometry and non-geometry information in NASA-IGES files in a high-level, IGES-independent format. It also specifies the

database information which can easily be extracted from a NASA-IGES data file. Very little nongeometry product data is included, e.g., no material properties or surface finish properties. Currently, only surface models are included in this report.

In particular, this specification *does*:

- describe the geometry and other information available to CFD grid generation software through NASA-IGES files.

- describe an *abstract description of an in-memory, database* which contains complete information from NASA-IGES files.

- specify in Appendix A the specific IGES entities that are allowed in a NASA-IGES file and the recommended use and conversion of many IGES entities.

This specification *does not* :

- specify all the details of IGES that information is available in reference 1.

- specify any on-disk or in-memory database or any specific database formats.

- specify an implementation of the abstract database.

It is expected that this specification will be updated and improved to support future enhancements to IGES, future needs of the research community, and to correct any problems that arise. Some specific issues that are to be addressed in the future include:

- inclusion of the Solid Modeling Boundary-Representation (B-Rep) capability. This capability is needed for advanced grid generation software.

- exchange of formats for structured grids, unstructured grids, and solution data.

- expansion to include multidisciplinary research, including structures, propulsion, and controls.

1.4 NASA Support

This specification has the direct support of the NASA Surface Modeling and Grid Generation Steering Committee representing the NASA Headquarters Office of Aerospace Science and Technology (OAST), three NASA Research Centers: Ames, Langley, and Lewis, and two operational NASA facilities: Johnson Space Center and Marshall Space Flight Center. The contributors include over 15 engineers and scientists from the three research centers, only some of whom can be listed here.

Christopher M. Cagle	NASA Langley Research Center
Melinda F. Cagle	NASA Langley Research Center
Jeffery A. Cerro	Lockheed Engineering and Sciences Company (NASA Langley)
Kevin Downey	Sverdrup Inc. (NASA Lewis)
Francis Y. Enomoto	NASA Ames Research Center
Austin Evans	NASA Lewis Research Center
Raymond C. Luh	MCAT Institute (NASA Ames)
Matthew Melis	NASA Lewis Research Center
Harold Renkel	NASA Lewis Research Center
Robert P. Weston	NASA Langley Research Center

These NASA facilities are committed to utilizing this specification for geometry representation for design and analysis of aerospace vehicles utilizing CFD techniques.

2.0 The CFD Process

NASA research centers support studies in a variety of scientific areas. Utilizing computer simulation, NASA supports extensive research on analysis of the behavior of complex physical fields. Examples of physical field analysis include computational fluid dynamics (CFD), computational electro-magnetics (CEM), heat transfer, and finite element modeling (FEM). Virtually any field that utilizes partial differential equations (PDE) performs some form of field solution calculation. Most of these fields study the effects of a phenomenon around a particular object. The numerical data that provide a mathematical description of that object are called the geometry data or model data. For these computer simulations to be useful in the design process, the geometry data must be passed between many groups rapidly and accurately. Even in the case of pure research the geometry data must be shared with other groups. For example, in a typical fluid dynamics study, the computational solutions are compared to wind tunnel data. The manufacturer of the wind tunnel model must have an accurate geometry definition from the computational scientist.

This specification addresses the geometry data transfer and geometry data usage requirements for these complex field simulations. The specific research area most applicable at this time is CFD. The remainder of this section focuses mainly on the CFD process but is generally applicable to other field simulation processes.

2.1 The CFD Analysis Process

Research in CFD is accomplished by modeling the fluid as a discrete set of points and computing the velocity and other properties of the fluid at each of these points. The set of points is referred to as a "grid" or "mesh."

A general representation of the CFD analysis process and the data transferred is shown in figure 1.

Two forms of geometry definition are utilized by grid generation tools: surface definition and solid definition.

In surface definition, the surfaces of the object to be analyzed are required by the grid generation tool. Currently, a majority of the grid generation tools require this surface geometry definition. Most of the grid generation tools utilizing surface definition are used interactively in the grid generation process.

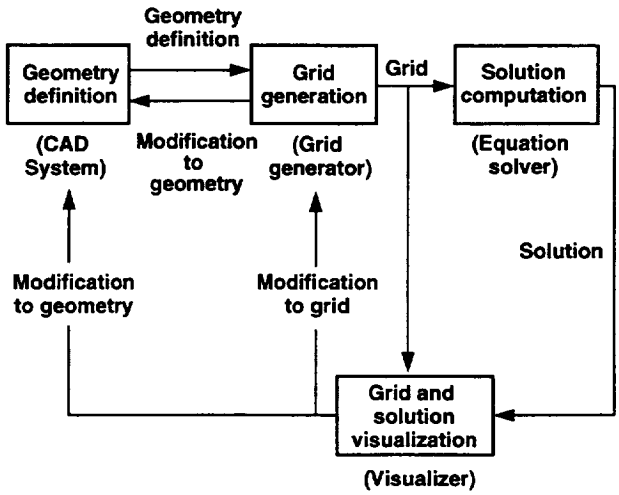


Figure 1. CFD analysis process.

Solid definition is usually required by tools that perform automatic grid generation. Currently, only a few grid generation tools utilize solid geometry definition. This specification is intended for surface geometry only; future enhancements may include solid definition.

2.2 The CFD Design Process

When CFD is used for design, the analysis process must be done a great many times in order to determine an optimal design.

If a computer-aided design (CAD) system is available to the analysis group, the geometry is modified on the CAD system and the new data are transferred to the grid generator and the grid generation process is repeated. This "pipeline" is repeated for each different design modification. Even though most current CAD systems can provide geometry in IGES format today, transferring new geometry for each analysis has to be done via data conversion, since grid generation tools do not currently read IGES data. This conversion is tedious and usually introduces additional errors.

CAD systems are often not available to the analysis group. In these cases, the grid is modified and the rest of the grid generation process is repeated. Current tools perform this geometry modification through changes to the grid representing the surface of the model. Such tools are limited in their capabilities and introduce additional data errors. Once a particular configuration is selected, the grid representing the surface of the model is transferred to the CAD system for re-integration with the original model. The methods for this data transfer are not standardized

and the data will have to be converted into the particular CAD system's format. This entire process is tedious and usually introduces additional data conversion errors.

2.3 Problems with Current Methods

The current methods for data transfer and grid generation are very time- and manpower-intensive and often require data approximation during conversion and use.

Currently, most grid generation software cannot utilize the geometry definition generated by CAD systems directly. The geometry has to be massaged into the different ad hoc formats required by the different types of grid generation software, and there are as many formats as there are grid generation packages. These formats frequently utilize only discrete point information for representing the geometry and do not retain complete information about the geometry.

Intensive human interaction and extensive manipulation are commonly required to convert the geometry into the particular format required by a piece of grid generation software. These operations are laborious and error-prone. The geometric information, e.g., surface curvatures, lost during this conversion is either extremely difficult or impossible to recover. Sometimes, the incompleteness of geometric information imposes severe limitations on the capability of the grid generation software.

Consistent utilization of NASA-IGES would dramatically improve these areas.

2.4 CFD Design Utilizing NASA-IGES (Initial Graphics Exchange Specification)

If both the geometry definition system and the grid generator can utilize the NASA-IGES Specification data format without any conversion errors, geometry data can be passed back and forth quickly and accurately. A series of design modifications could be generated on a CAD system and transferred to the grid generation software in minutes. The first configuration may require a fair amount of time to perform surface gridding, volume gridding, and solution computation. Successive iterations should be available in very little time if the gridding programs can rapidly regenerate new grids from new geometry data that has similar topology to the previous data. The errors identified in section 2.3 could be eliminated entirely if both the CAD system and the grid generation software operate on the same geometry data.

The NASA-IGES specification is designed to be bidirectional. Software systems should be capable of both reading and writing data in this NASA-IGES format. This will require grid generation programs to read in NASA-IGES data, perform any modifications directly on the NASA-IGES geometry rather than the computational grid, and to write out modified surfaces in NASA-IGES format.

2.5 CFD Design Utilizing the Supplied Database Information Format

To facilitate the use of this data transfer method, NASA is developing software for several functions such as reading, writing, and translating NASA-IGES data. For more information on these programs, contact:

Matthew Blake
MS T27B-2
NASA Ames Research Center
Moffett Field, CA 94035-1000

Utilizing these programs and their implementation of the abstract database described herein, grid generation software can utilize NASA-IGES geometry through their existing in-memory database without handling NASA-IGES files directly. One possible scheme for such in-memory data access is through a shared memory architecture utilizing the reader-writer software. Alternatively, the user may choose to utilize NASA-IGES data files for transfer, using this document for an understanding of the mathematics behind the geometric entities while using the IGES document to understand the file format.

Since different grid generators may require different internal database formats to satisfy individual needs, a grid generator may need to convert the in-memory database obtained from the reader-writer to its internal in-memory database format. The process for a grid generator to use the reader-writer is expressed in figure 2. All the stages in the process can be done internally and automatically by the grid generator, and the reader-writer will do most of the work of incorporating NASA-IGES files.

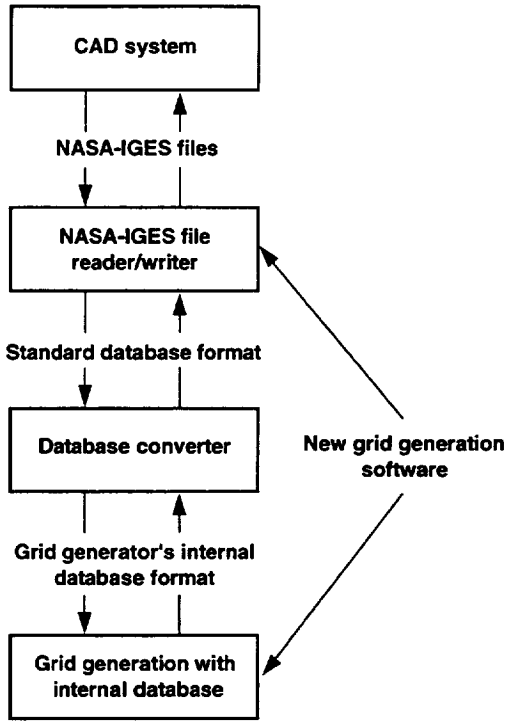


Figure 2. Grid generation with NASA-IGES file reader-writer.

3.0 General Information on Data Description

The geometry and nongeometry information defined in this document is separated into logical units, each of which is called an *object* or an *entity*. The word *entity* is used in this document. An entity represents either a complete geometric concept or a complete bit of information. However, in some instances an *entity* becomes meaningful in the database only after it is attached to another entity. The syntax and semantics of several methods for performing such attachment is described throughout this document.

3.1 Entity Description Overview

The geometry entities are defined in section 4; the nongeometry entities are defined in section 5. The IGES entity type number for the entity is included in parentheses by the entity's name. For each entity, a general description is first given, followed by a mathematical definition and/or more detailed information. Illustrations are provided when appropriate. These descriptions are very abstract and hence do not depend on the exact data stored for the entity.

For each entity, a Database Information subsection is provided. Collectively, these subsections and section 6 define an abstract representation of an in-memory database. Even though the information listed under these Database Information subsections closely resembles the data specified in the IGES file format, this specification does not specify an implementation of the abstract database, the exact stored data, individual data types and data structures, or the specific storage formats. However, this specification requires an implementation to provide methods or routines to extract the information defined in the abstract database. For example, an implementation shall provide methods for its applications to obtain the coefficients of a conic (section 4.3.4) regardless of how the conic is stored internally in the specific computer program implementation. The reason for not specifying the exact data format is that the data format depends heavily on the computer language used in an implementation. For example a two-dimensional (2D) array may be best for illustrating data in Fortran, while an array of pointers may be best in the C language.

There are two reasons for the close resemblance between the abstract database and the IGES format. First, it is easier for the reader-writer to set up the database. Second, complete information will be easily preserved during the file-to-database and database-to-file conversion. An implementation of this database format can be found in the NASA-IGES view document mentioned in section 2.5.

The word "array" used in the description denotes an ordered list of items. The same item can appear more than once in an array. The array can be implemented by a data array structure in a computer language, but this specification does not imply such practice.

Finally, comments are provided in a Note subsection for most entities.

An entity may have different *form*s, denoting different interpretations. Both numbers and tokens are used in this specification to denote the values of forms, but, again, this specification does not imply such practices in an implementation. The storage of the form is discussed in section 6 (also see section 3.3).

3.2 Coordinate System

All of the entities are defined in a local coordinate system which forms the *definition space* of the entity. The local coordinate system is usually the most convenient and stable coordinate system to define the entity. However, the designed model usually resides in a different coordinate system, called the *model space*. The local coordinate system may coincide with the model space coordinate system. If not, one or more coordinate transformation matrices must be used to bring the entity from its definition space position to its final model space position.

A model may be designed at an enlarged or reduced size. To obtain its real world size, the dimensions of a model as specified in the database must be divided by the factor Model Space Scale (section 6.2).

A *transformation matrix pointer* is associated with every entity. This pointer is either 0, for the identity rotation matrix and zero translation vector, or a transformation matrix entity which shall be applied to the entity in the process of bringing the entity to the model space. (In fact, a transformation matrix entity contains a transformation matrix pointer. Hence, it is possible to store successive transformations under one transformation matrix pointer. See section 5.1 for more information.)

Since the database is hierarchical, i.e., an entity may be a part of another entity, recursively, multiple transformation matrices, following the hierarchy, may be necessary to bring an entity from its definition space to the model

space. For example, if entity A is a part of the definition of entity B, entity A shall be transformed by the transformation matrix associated with A first and then by that associated with B.

All the coordinate systems are right-handed.

3.3 Common Information

Information common to all entities is not described under the database information subsection for each individual entity. This information includes, but is not limited to, color, level, form, and transformation matrix pointer.

Some information common to the entire model and data files is also contained in the database. This includes, but is not limited to, text identifying the model, measuring system units, and data of file creation. This information corresponds to the global section of the IGES files.

The common and global information and other database related issues are discussed in section 6.

4.0 Geometric Entities

4.1 Circular Arc or Circle (IGES Entity Type Number 100)

4.1.1 Circular Arc: General Description

A circle is a 2D curve whose points are at a constant distance from a point. A circular arc is a connected set of points on a circle. Both a circle and a circular arc shall consist of more than one point.

4.1.2 Circular Arc: Mathematical Formulation

A circle or circular arc always lies in a constant Z-plane in its definition space. Transformation matrices shall be used if it is necessary to bring the circle or circular arc to the proper position in the model space.

A circular arc can be defined by a start point (X_2,Y_2), a terminate point (X_3,Y_3), and an arc center point (X_1,Y_1). The arc goes from the start point to the terminate point counterclockwise around the positive Z-direction in its definition space. Hence, the order of the end points (start and terminate) distinguishes an arc from its complementary arc.

$$C(t) = (X_1 + R * \mathrm{Cos}(t), Y_1 + R * \mathrm{Sin}(t), Z_1)$$

$$\text{for } t_2 <= t <= t_3$$

where, for i = 2 and 3

$$R = \sqrt{(X_i - X_1)^2 + (Y_i - Y_1)^2}$$

t_i is such that $(R^* \mathrm{Cos}(t_i), R^* \mathrm{Sin}(t_i)) = (X_i - X_1, Y_i - Y_1)$

and $0 <= t_2 < 2^* \pi$, $0 <= t_3 - t_2 <= 2^* \pi$

4.1.3 Figure 3: Circular Arc

Figure 3 shows a circular arc on the X-Y plane (Z = 0).

4.1.4 Circular Arc: Database Information

The database contains the following information for a circle.

Z — The parallel Z displacement of arc from the X-Y plane

X_1 — The arc center abscissa

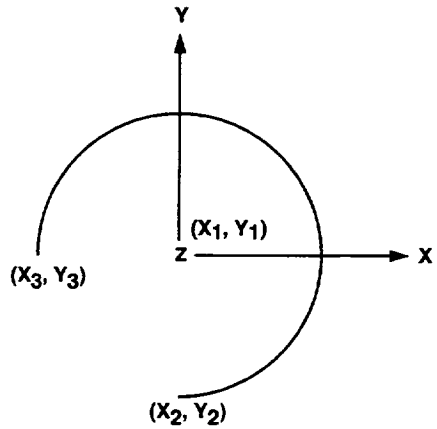


Figure 3. Circular arc.

Y_1 — The arc center ordinate

X_2 — The start point abscissa

Y_2 — The start point ordinate

X_3 — The terminate point abscissa

Y_3 — The terminate point ordinate

4.1.5 Circular Arc: Note

By definition, a circle or circular arc can never be a point. A circle or circular arc can be represented by a rational B-spline curve exactly. However, the above representation is more compact. In addition, the extraction of some crucial information, like the radius, from a rational B-spline representation is numerically unstable.

4.2 Composite Curve (IGES Entity Type Number 102)

4.2.1 Composite Curve: General Description

A composite curve is a continuous curve composed of either a single curve or an ordered list of curves (excluding a composite curve). Two-dimensional (2-D) and three-dimensional (3-D) curves are both acceptable.

A composite curve is a directed curve, having a start point and a terminate point. The direction of the composite curve is indicated by the order of the constituent curves. The composite curve goes from the first constituent curve to the second, to the third, and so on. Within the defining list, the terminate point of each constituent curve has the

same coordinates as the start point of the succeeding curve.

4.2.2 Composite Curve: Mathematical Formulation

A composite curve defined by a list of parametric curves can be written as follows:

Let

1. N be the number of parametric curves in the composite curve; $CC_i(t)$, for $1 <= i <= N$, be the ordered parametric curves with parameter t in $[ps_i, pe_i]$, where ps_i and pe_i are the start and end parameters of curve i;

2. $t_0 = 0$

3. $t_i = t_{i-1} + (pe_i - ps_i)$ for each i in [1, N]

Then

1. $CC_i(pe_i) = CC_{i+1}(ps_{i+1})$, for $1 <= i <= N-1$

2. The composite curve $C(t) = CC_i(t - t_{i-1} + ps_i)$, where
   $t_{i-1} <= t <= t_i$, for $1 <= i <= N$ and
   t is the global parameter of the composite curve.

4.2.3 Figure 4: Composite Curve

Figure 4 is a composite curve consisting of a circular arc, a straight line segment and an elliptical arc.

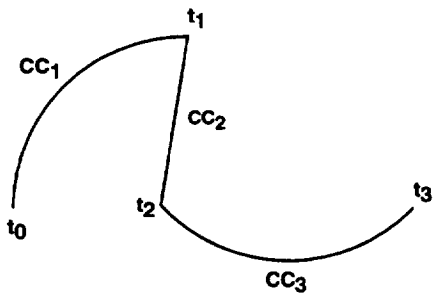


Figure 4. Composite curve.

4.2.4 Composite Curve: Database Information

The database contains the following information for a composite curve.

nCurve — The number of curves.

curves — An array of pointers to the constituent curves. The curves in the array are ordered, starting from the first curve.

4.2.5 Composite Curve: Note

A composite curve can also be a closed curve, i.e., the first and last points of the curve coincide.

When a circular arc or a conical arc appears in a composite curve, the start point of the arc, as specified in the database, must be the first point of the arc in the composite curve. Since an arc is defined counterclockwise in its local coordinate system, in some cases it is necessary to adjust the local coordinate system of an arc to meet the above requirement. For example, in figure 3, the lower-left point of the circular arc is above the start point of the composite curve. However, in order for the lower-left point to be the start point of the arc and for the arc to be defined counterclockwise, the z-direction of the local coordinate system of the arc must be shown pointing into the page.

4.3 Conic Arc (IGES Entity Type Number 104)

4.3.1 Conic Arc: General Description

A conic arc is a bounded, connected portion of a parent conic curve consisting of more than one point (of finite length). The parent conic curve is either an ellipse, a parabola, or a hyperbola. The curve is always defined on a constant Z-plane in its definition space. A transformation matrix should be used to bring the curve to its model space position if necessary.

4.3.2 Conic Arc: Mathematical Definition

A conic on the X–Y–plane is defined by the six coefficients (A–F) in the following equation.

$$A X^2 + B X Y + C Y^2 + D X + E Y + F = 0$$

Each coefficient is a real number. A conic arc has two endpoints. By considering the endpoints as ordered, i.e., a start point and a terminate point, a direction can be associated with the arc.

In order to distinguish the desired elliptical arc from its complementary arc, the direction of the desired elliptical arc is defined to be counterclockwise around the Z coordinate axis in its definition space.

The definitions of the terms *ellipse*, *parabola*, and *hyperbola* are given in terms of the quantities Q1, Q2, and Q3, where

$$Q1 = \text{determinant of} \begin{bmatrix} A & B/2 & D/2 \\ B/2 & C & E/2 \\ D/2 & E/2 & F \end{bmatrix}$$

$$Q2 = \text{determinant of} \begin{bmatrix} A & B/2 \\ B/2 & C \end{bmatrix}$$

$$Q3 = A + C$$

A parent conic curve is:

an ellipse if $Q2 > 0$ and $Q1 * Q3 < 0$,

a hyperbola if $Q2 < 0$ and $Q1 \neq 0$, or

a parabola if $Q2 = 0$ and $Q1 \neq 0$.

Please refer to Thomas (ref. 2) for specific details.

4.3.3 Figure 5: Conic Arc

The following are figures showing some conic arcs.

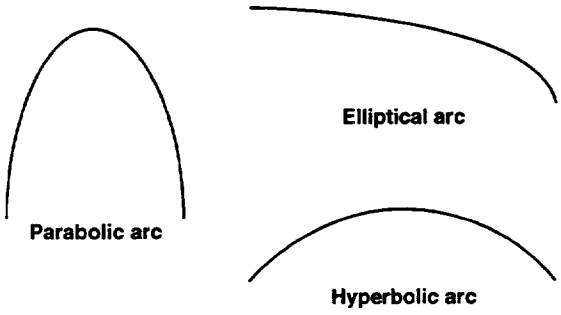


Figure 5. Conic arc: several examples.

4.3.4 Conic Arc: Database Information

The database contains the following information for a conic.

A,B,C,D,E,F	The coefficients of the conics
Z	The Z-coordinate of the plane of definition
X_1	The start point abscissa
Y_1	The start point ordinate
X_2	The terminate point abscissa
Y_2	The terminate point ordinate

4.3.5 Conic Arc: Note

A full ellipse is represented by an elliptical arc with coincident endpoints.

Conic arcs as specified are extremely sensitive to the values of the coefficients. Small changes in the values may cause dramatic changes in curve shapes. Determination of the curve type from the coefficients is also very unstable for most cases. Please refer to Appendix C of IGES Version 5.1 (ref. 1) for more information.

Conic arcs can be represented exactly by rational B-spline curves, which provide a stable method for determining points on the curves and curve shapes. However, extracting geometry information, such as the major and minor axis, from the rational B-spline representation may be unstable.

4.4. Copious Data (IGES Entity Type Number 106, Forms 1, 2, 3)

4.4.1 Copious Data: General Description

"Copious data" is a list of points.

4.4.2 Copious Data: Mathematical Definition

There are three forms of copious data, described below. In the following formulation, N is the number of points; X_i, Y_i, Z_i are the coordinates of point P_i; V_{xi}, V_{yi}, and V_{zi} are the components of a vector associated with point P_i.

Form 1

The data points are on a constant Z-plane in the definition space. The points can be represented as

$$(X_0, Y_0), \ldots, (X_N, Y_N), Z$$

where Z is the Z-coordinate value of the constant Z-plane.

Form 2

The data points are in the three-dimensional definition space (may have different Z-coordinates). The points can be represented as

$$(X_0, Y_0, Z_0), \ldots, (X_N, Y_N, Z_N)$$

Form 3

The data points are in the definition space. In addition to the coordinates, a three-space vector is also associated with each point. The data can be represented as

$(X_0, Y_0, Z_0, V_{x0}, V_{y0}, V_{z0}), \ldots, (X_N, Y_N, Z_N, V_{xN}, V_{yN}, V_{zN})$. The meaning of the vector is not specified by this document, so the user can interpret it for each case.

4.4.3 Figure 6: Copious Data

An example of copious data is shown in figure 6.

4.4.4 Copious Data: Database Information

interpFlag — The interpretation flag

- 1: Form 1. The data points are on a constant Z plane in (x, y) pairs.
- 2: Form 2. The data points are in three-dimensional space in (x, y, z) triples.
- 3: Form 3. The data points are in three space with a three-space vector at each point, in (x, y, z, vx, vy, vz) sextuples.

ZT — The common Z coordinate of the points, valid only for Form 1.

number — The number of data points.

dataPoints — An array of data points. Each point is stored as a pair, triple, or sextuple for Forms 1, 2, and 3, respectively.

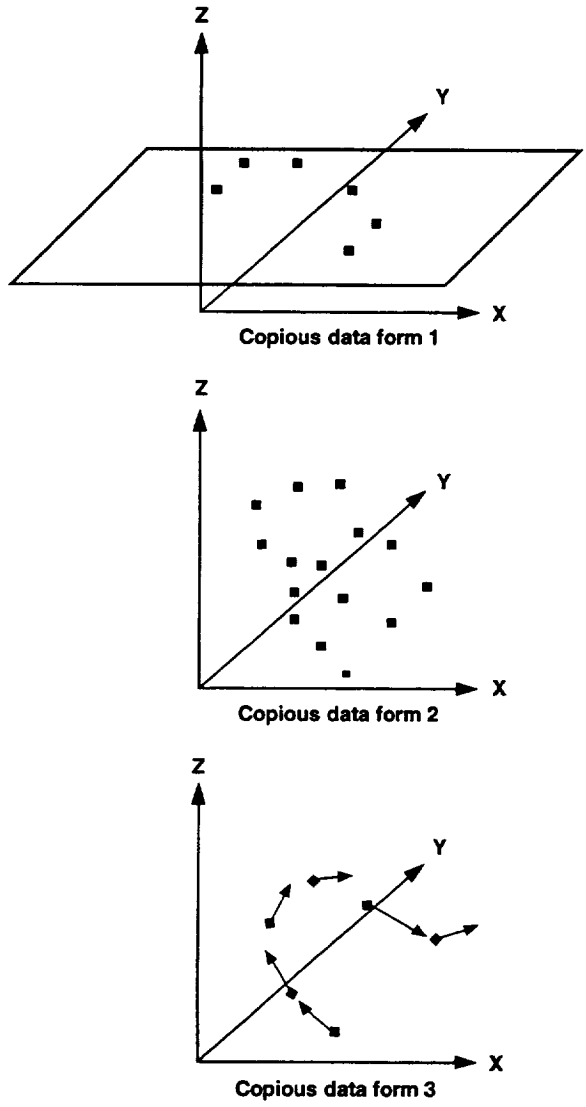


Figure 6. Copious data.

4.4.5 Copious Data: Note

Copious data are used to represent points on cross-sections.

4.5. Line
(IGES Entity Type Number 110)

4.5.1 Line: General Description

This is a line segment (bounded line) with more than one point (of finite length).

4.5.2 Line: Mathematical Formulation

A line with end points P_1 and P_2 can be written as

$L(t) = (1 - t)P_1 + tP_2$, for $0 <= t <= 1$.

4.5.3 Figure 7: Line

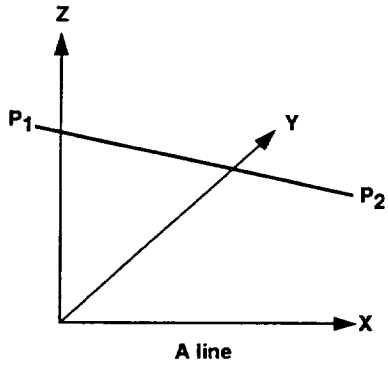


Figure 7. Line.

4.5.4 Line: Database Information

The database contains the following information for a line.

X_1 The start point X-coordinate

Y_1 The start point Y-coordinate

Z_1 The start point Z-coordinate

X_2 The terminate point X-coordinate

Y_2 The terminate point Y-coordinate

Z_2 The terminate point Z-coordinate

4.5.5 Line: Note

A line segment can also be represented as a B-spline curve. However, this line entity is more compact.

4.6. Point (IGES Entity Type Number 116)

4.6.1 Point: General Description

A point is a point in space.

4.6.2 Point: Mathematical Formulation

A point can be presented by its coordinates (X, Y, Z) in space.

4.6.3 Figure 8: Point

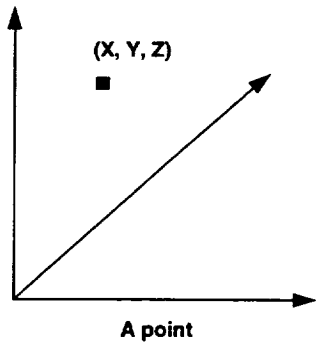


Figure 8. Point.

4.6.4 Point: Database Information

The database contains the following information for a point.

X The X-coordinate of the point

Y The Y-coordinate of the point

Z The Z-coordinate of the point

4.7 Rational B-Spline Curve (IGES Entity Type Number 126)

4.7.1 Rational B-Spline Curve: General Description

A rational B-spline curve is a piecewise parametric rational (or nonrational as a special case) polynomial curve with B-splines as basis functions. The knots that define the B-spline basis functions can be nonuniform.

Rational B-spline curves have been selected as the most stable format to represent all types of polynomial or rational polynomial parametric curves. With appropriate flags and knot vectors, rational B-spline curves are capable of representing single span or spline curves of explicit polynomial, rational, Bezier or B-spline curves. They can also represent most commonly used parametric curves and analytic curves exactly, e.g., parametric cubic spline curves and conics.

4.7.2 Rational B-Spline Curve: Mathematical Formulation

A NURBS curve is expressed in the form,

$$C(t) = \frac{\sum_{i=0}^{K} W(i)(P(i)b_{i,M}(t))}{\sum_{i=0}^{K} (W(i)b_{i,M}(t))}$$

where

W(i) are the weights (positive real numbers),

P(i) are the control points (points in R^3),

and $b_{i,M}(t)$ are the B-spline basis functions of degree M.

The real line represented by $t \in [-\infty, \infty]$ is the domain of the B-spline basis functions and is commonly referred to as the parameter space of the B-spline curve.

For each control point P(i), W(i) is the weight associated with the point. Once the B-spline basis functions are given, the shape of the curve is largely controlled by the control points, but the shape is also influenced by the weights.

The B-spline basis functions $b_{i,M}(t)$ are piecewise polynomial functions, determined by their degree, M, and their knot vector. The knot vector is a sequence of nondecreasing floating point numbers from the domain of the basis functions; they are the points where the values resulting from the pieces of the polynomials from the basis functions meet.

Let the knot vector be $t_0, \ldots, t_{N1}$. Where $N1 = M + K + 1$, the basis functions are defined as:

$$b_{i,M}(t) = \frac{t - t_i}{t_{i+M} - t_i} b_{i,M-1}(t) + \frac{t_{i+M+1} - t}{t_{i+M+1} - t_{i+1}} b_{i+1,M-1}(t)$$

where

$$b_{i,0}(t) = \begin{cases} 1, & \text{if } t \text{ in} [t_i, t_{i+1}) \\ 0, & \text{otherwise} \end{cases}$$

The curve is commonly defined in the domain interval $[t_M, t_{N1-M}]$. However, a subrange, [ts, te], of $[t_M, t_{N1-M}]$ can be used as the domain of the curve.

When the spacings of the knots in a knot vector are equal, i.e., Δt_i are the same, where $\Delta t_i = t_{i+1} - t_i$, $i = 0, \ldots, N_{1-1}$, the knot vector is called a uniform knot vector, and the B-splines are called uniform B-splines. In this case, the B-splines are related to each other by a translation in the domain space: $b_{i,M}(t) = b_{j,M}(t + (t_j - t_i))$. The curves with such a knot vector are called uniform B-spline curves.

When W(i) are the same for all i, the effect of the weights on the curve disappears. The curve can be written into the following form:

$$C(t) = \sum_{i=0}^{K} (P(i)b_{i,M}(t))$$

This curve is polynomial or nonrational.

When the first point and last point of the curve coincide, i.e., C(ts) = C(te), the curve is called a closed curve. Otherwise, it is called an open curve.

A curve is periodic if both the control points and knot vector satisfy the following constraints:

1. The last K control points are identical to the first K control points. That is, $P(i + M - K + 1) = P(i)$, for $i = 0, \ldots, K - 1$.

2. The last 2K knots $(t_{N1} - 2K + 1, \ldots, t_{N1})$ are related to the rest of the knots as defined in the following.

Let $\Delta t_j = t_{j+1} - t_j$, for $j = 0, \ldots, M - K$, and

$$\alpha_i = \sum_{j=0}^{i-(N1-2K+1)} \Delta t_{\text{mod}(j,M-K+1)}, \text{ for } i = N1 - 2K + 1, \ldots, N1$$

Then, $t_i = t_{N1-2K} + \alpha_i$, for $i = N1 - 2K + 1, \ldots, N1$.

A periodic curve is closed in the domain interval $[t_M, t_{N1-M}]$ with $C(t_M) = C(t_{N1-M})$.

The rational B-spline curve may represent analytic curves of general interest. The form information of the entity communicates the analytic curve types. The possible forms are: (1) undetermined, (2) line, (3) circular arc, (4) elliptic arc, (5) parabolic arc, and (6) hyperbolic arc.

When the form is undetermined, it may be that the curve is not any of the analytic curve types, or that the analytic curve type of the curve is not determined yet.

See Farin (ref. 3) and the Bibliography for more information about B-splines.

4.7.3 Figure 9: Rational B-Spline Curve

Figure 9 shows a B-spline curve.

Curve1 in figure 9 is a rational cubic (K = 3) B-spline curve with knot vector = {0., 0., 0., 0., .5, 1., 1., 1., 1.}. This curve is nonuniform, open, and rational if all the weights are not the same.

Curve2 is a nonrational cubic B-spline curve with knot vector = {0., 1., 2., 3., 4., 5., 6., 7., 8., 9., 10., 11.} and control points = {P(0), P(1), P(2), P(3), P(4), P(5), P(6), P(7)}, where P(5) = P(0), P(6) = P(1), and P(7) = P(2). The curve is closed, periodic and nonrational.

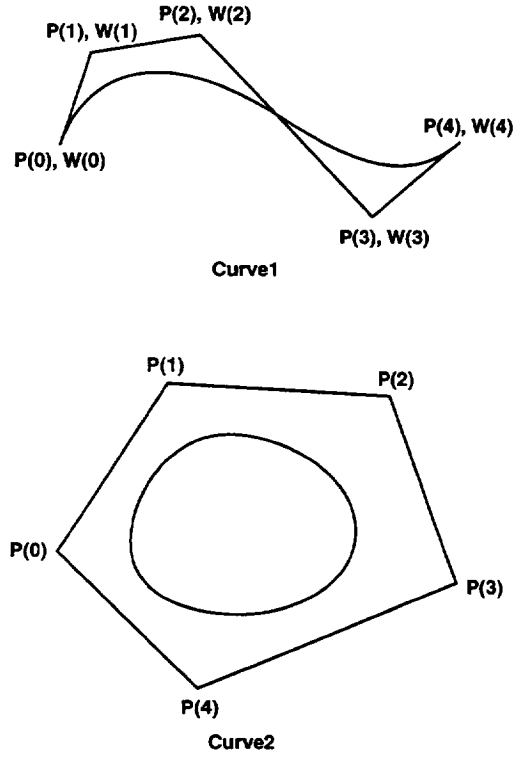


Figure 9. Rational B-spline curve.

4.7.4 Rational B-Spline Curve: Database Information

The database contains the following information for a rational B-spline curve.

K	The number of control points minus one.
M	The degree of the B-spline
planar	A flag to indicate whether the curve is completely on a plane
closed	A flag to indicate whether the curve is closed or open
periodic	A flag to indicate whether the curve is periodic or nonperiodic
rat_or_poly	A flag indicating whether the curve is rational or polynomial
t	The array of knot vectors, in non-decreasing order
W	The array of weights
P	The array of control points. Each point has three components: X, Y, and Z.
ts	The start parameter of the curve
te	The end parameter of the curve
unit_normal	The unit normal of the plane which the curve is on, valid only if the curve is planar

4.7.5 Rational B-Spline Curve: Note

The weights of a B-spline curve have to be positive.

4.8 Rational B-Spline Surface (IGES Entity Type Number 128)

4.8.1 Rational B-Spline Surface: General Description

A rational B-spline surface is a piecewise, rational, polynomial surface. A rational B-spline surface is also the projection to 3-D Euclidean space of a tensor product nonrational B-spline surface defined in 4-D homogeneous space. The basis functions of a tensor product nonrational

B-spline surface are the Cartesian product of the basis functions of two B-spline curves.

Nonrational B-spline surfaces are a subset of rational B-spline surfaces. In addition, a rational B-spline surface can represent various commonly used analytical surfaces, e.g., cone, cylinder, and torus.

4.8.2 Rational B-Spline Surface: Mathematical Formulation

A rational B-spline surface is expressed in the form,

$$S(u,v) = \frac{\sum_{i=0}^{K_1}\sum_{j=0}^{K_2}(W(i,j)P(i,j)b_{i,M1}(u)b_{j,M2}(v))}{\sum_{i=0}^{K_1}\sum_{j=0}^{K_2}(W(i,j)b_{i,M1}(u)b_{j,M2}(v))}$$

where

W(i,j) are the weights (positive real numbers),

P(i,j) are the control points (points in R3),

and $b_{i,M1}(u)$ and $b_{j,M2}(v)$ are the B-spline basis functions.

The plane formed by the Cartesian product of u and v, [u, v], is commonly referred to as the parameter space of the surface.

P(i,j), for $0 <= i <= K_1$, $0 <= j <= K_2$, is a topologically rectangular mesh of points. For each point P(i,j), W(i,j) is the weight associated with the point. Once the B-spline basis functions are given, the shape of the surface is controlled mainly by P(i,j) and adjusted by W(i,j).

Each of the $b_{i,M1}(u)$ and $b_{j,M2}(v)$ is the basis function for a B-spline curve. Let the knot vector for $b_{i,M1}(u)$ be $u_0, \ldots, u_{N1}$ and the knot vector for $b_{j,M2}(v)$ be $v_0, \ldots, v_{N2}$, where $N1 = M1 + K_1 + 1$, $N2 = M2 + K_2 + 1$. $b_{i,M1}(u)$ and $b_{j,M2}(v)$ are defined as:

$$b_{i,M1}(u) = \frac{u - u_i}{u_{i+M1} - u_i} b_{i,M1-1}(u) + \frac{u_{i+M1+1} - u}{u_{i+M1+1} - u_{i+1}} b_{i+1,M1-1}(u)$$

where

$$b_{i,0}(u) = \begin{cases} 1, \text{ if } u \text{ in } [u_i, u_{i+1}) \\ 0, \text{ otherwise} \end{cases}$$

and

$$b_{j,M2}(v) = \frac{v - v_j}{v_{j+M2} - v_j} b_{j,M2-1}(v) + \frac{v_{j+M2+1} - v}{v_{j+M2+1} - v_{j+1}} b_{j+1,M2-1}(v)$$

where

$$b_{j,0}(v) = \begin{cases} 1, \text{ if } v \text{ in } [v_j, v_{j+1}) \\ 0, \text{ otherwise} \end{cases}$$

The surface is commonly defined in the subspace, $[u_{M1}, u_{N1-M1}] X [v_{M2}, v_{N2-M2}]$, of the parameter space of the surface. However, a subrange, [us, ue] X [vs, ve], of $[u_{M1}, u_{N1-M1}] X [v_{M2}, v_{N2-M2}]$ can be used as the domain of the surface.

The rational B-spline surface is uniform in the u direction if the u knot vector, $u_0, \ldots, u_{N1}$, is uniform (see section 4.7.2 for the definition of a uniform knot vector). The same is true for the v direction.

The rational B-spline surface is closed in the u direction if S(us,v) = S(ue,v), for all v in [vs, ve]. The rational B-spline surface is closed in the v direction if S(u,vs) = S(u,ve), for all u in [us, ue].

The rational B-spline surface is periodic in the u direction if

1. The last K control points of each row in the u direction are identical to the first K control points of each row in the u direction. That is, $P(i + M1 - K_1 + 1, j) = P(i,j)$, for $i = 0, \ldots, K_1 - 1$ and for all j.

2. The last $2K_1$ knots ($u_{N1-2K1+1}, \ldots, u_{N1}$) are related to the rest of the knots as defined in section 4.7.2.

A periodic surface in the u direction is closed in the domain interval $[u_M, u_{N1-M}]$ with $S(u_M,v) = C(u_{N1-M},v)$, for all v.

Similar properties are true when the rational B-spline surface is periodic in the v direction.

When W(i,j) are the same for all i and j, the effect of the weights on the surface disappears. The surface is polynomial or non-rational.

The rational B-spline surface may represent analytic surfaces of general interest. The form information of the entity communicates the analytic surface types. The possible forms are: (1) undetermined, (2) plane, (3) right circular cylindrical patch, (4) right circular conical patch, (5) spherical patch, (6) toroidal patch, (7) surface of revolution, (8) ruled surface, and (9) general quadric surface.

When the form is undetermined, it may be that the surface is not any of the analytic surface types, or that the analytic surface type is not determined yet.

See Farin (ref. 3) and the Bibliography for more information about B-splines.

4.8.3 Figure 10: Rational B-Spline Surface

The following is a rational B-spline surface with K1 = 4, K2 = 2, M1 = 3, M2 = 2, a knot sequence in the u direction = (0., 0., 0., 0., .5, 1., 1., 1., 1.), and a knot sequence in the v direction = (0., 0., 0., 1., 1., 1.).

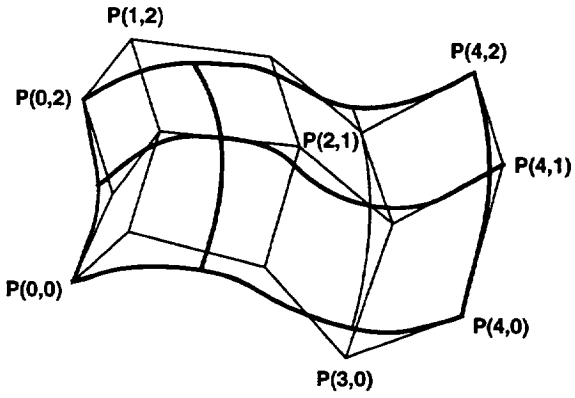


Figure 10. Rational B-spline surface.

4.8.4 Rational B-Spline Surface: Database Information

The database contains the following information for a rational B-spline surface.

K1	The number of control points in the first direction minus one
K2	The number of control points in the second direction minus one
M1	The degree of the B-splines in the first direction
M2	The degree of the B-splines in the second direction
closed_u	A flag to indicate whether the surface is closed or open in the first direction
closed_v	A flag to indicate whether the surface is closed or open in the second direction
periodic_u	A flag to indicate whether the B-splines are periodic in the first direction
periodic_v	A flag to indicate whether the B-splines are periodic in the second direction
rat_or_poly	A flag to indicate whether the surface is rational or polynomial
u_knots	The array of knots in the first direction in nondecreasing order
v_knots	The array of knots in the second direction in nondecreasing order
W	The array of weights
P	The array of control points. Each point has three components: X, Y, and Z.
us	The starting parameter value in the first direction
ue	The ending parameter value in the first direction
vs	The starting parameter value in the second direction
ve	The ending parameter value in the second direction

4.8.5 Rational B-Spline Surface: Note

The weights of a rational B-spline surface must be positive.

4.9 Boundary (IGES Entity Type Number 141)

4.9.1 Boundary: General Description

A boundary entity is a closed, simply-connected curve on a surface, used primarily in the bounded surface entity to denote a portion of the underlying surface of concern.

A boundary entity partitions the underlying surface into two parts: one inside the boundary entity, the other outside the boundary entity. A convention, described below, allows one part to be designated as the portion of concern (or the active region).

4.9.2 Boundary: Mathematical Formulation

A boundary entity consists of a surface, a list of curves on the surface in the model space, and a corresponding list of curves in the domain space of the surface. The curves on the surface are called model space curves. The curves in the domain space of the surface are called parameter space curves.

4.9.2.1 Boundary: Surface

The surface must be a rational B-spline surface, denoted S(u,v). Let the domain of the surface be D = [a, b] X [c, d] (section 4.9.3, fig. 11(a)). The following is defined:

1. The *natural boundary* of the surface indicates the four curves (section 4.9.3, fig. 11(b)):

S(a,v), c <= v <= d;

S(b,v), c <= v <= d;

S(u,c), a <= u <= b;

S(u,d), a <= u <= b.

2. Let P be a 3-D model space point. Then P is a pole of the surface defined by mapping S(u,v) if any of the following are true:

P = S(a,v) for all v such that c <= v <= d

P = S(b,v) for all v such that c <= v <= d

P = S(u,c) for all u such that a <= u <= b

P = S(u,d) for all u such that a <= u <= b

The cone in section 4.9.3, figure 11(b) has a pole in its apex.

3. Let C be a model space curve. Then C is a seam of the surface defined by the mapping S(u,v) if it is the image in model space of

C(v) = S(a,v) for all v such that c <= v <= b and
C(v) = S(b,v) for all v such that c <= v <= d , or

C(u) = S(u,c) for all u such that a <= u <= b and
C(u) = S(u,d) for all u such that a <= u <= b.

The cone in section 4.9.3, figure 11(b) has a seam curve C(v) = S(a,v) = S(b,v), for c <= v <= d.

The surface in the boundary entity has to satisfy the following:

1. The mapping S(u,v) = (x(u,v), y(u,v), z(u,v)) is defined for each pair (u,v), where a <= u <= b and c <= v <= d. This requires the surface to be position-continuous.

2. The mapping is one-to-one in the interior of D. This disallows self-penetrating surfaces. However, any point on the natural boundary of the surface may coincide with any other point on the natural boundary of the surface. In particular, seams are allowed.

3. The surface must have continuous normal vectors at every point of domain D except those which map to poles. This requires that a unique tangent plane exists at every point where natural boundaries coincide, except at poles.

4.9.2.2 Boundary: Model Space Curve

Each of the model space curves, denoted as C_i, i = 1, n, has to satisfy the following:

1. The curve must have a unique nonzero tangent vector at each point.

2. The curve must lie on the surface (to within the accuracy of the computer system).

3. The curve may only self-intersect at its endpoints.

The direction of each individual model space curve may need to be reversed before its use in defining a boundary. The orientation of C_i has to satisfy the requirements given below. However, instead of reversing the curve representation, the abstract database has an orientation flag which indicates this reversion while C_i are kept in their original orientation.

Let G_i be the oriented C_i. G_i must satisfy the following (section 4.9.3, figs. 11(c)–11(e)):

1. The list of curves, G_i, i = 1, n, must form a closed loop. That is, the terminate point of G_n is the start point of G_1.

2. The terminate point of curve G_{i-1} is the start point of curve G_i, i = 2, n.

3. The list of curves do not self-intersect except at the start point of G_1 and the terminate point of G_n.

4.9.2.3 Boundary: Active Region Definition

In addition, the orientation of the list of curves, G_i, also defines the active region as follows:

1. The positive surface normal is given by the cross product (in the order specified) of the partial derivative of S(u,v) with respect to u, denoted S,u(u,v), and the partial derivative of S(u,v) with respect to v, denoted S,v(u,v) (section 4.9.3, fig. 11(b)).

2. The left side of a point, p, on G_i is the direction of the vector formed as the cross product (in the order specified) of the surface normal and the tangent vector to G_i at p (section 4.9.3, fig. 11(b)).

3. The active region is the portion of the surface enclosed by G_i and is to the left side of G_i.

The active region also must satisfy the following:

1. The active region has finite area.

2. Any two points on the active region must be path connected.

3. The interior of the active region lies on the left of G_i.

4. The active region consists of all of the points on its boundary, G_i, and its interior.

5. The closure of the interior of the active region (in the relative topology of the surface induced by R3) is the active region.

4.9.2.4 Boundary: Parameter Space Curve

For each model space curve C_i, there is a set of corresponding parameter space curves, F_{ij}, j = 1, m, defined below.

Each F_{ij} must satisfy the following (section 4.9.3, fig. 11(g)):

1. The curve must have a unique nonzero tangent vector at each point.

2. The curve must not self-intersect except, possibly, at its endpoints.

3. Let C^*_{ij} be the curve obtained by mapping F_{ij} onto the surface S, i.e.,

$C^*_{ij} = S \cdot Fij$

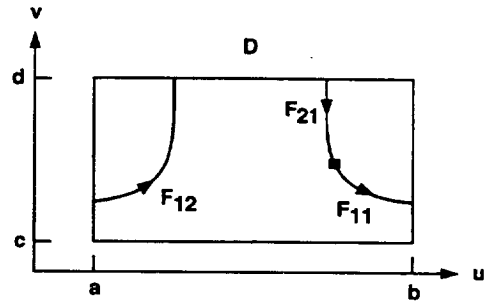
C^*_{ij} j = 1, m, must form a composite curve, denoted as C^*_i.

4. The composite curve C^*_i must coincide with C_i to within the accuracy of the computer system.

5. C^*_i must have the same orientation as C_i.

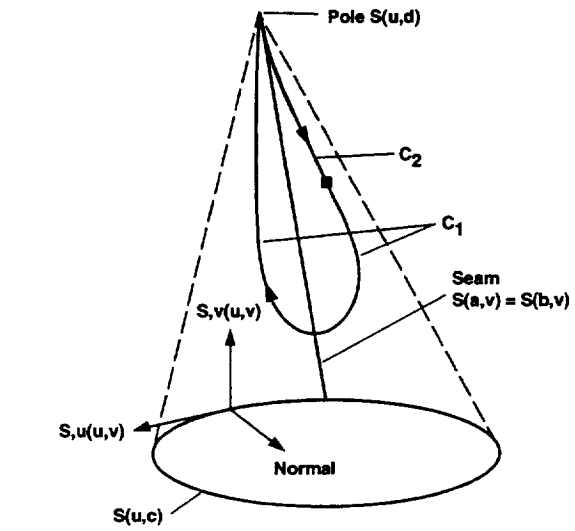
However, the collection of parameter space curves F_{ij}, i = 1, n, j = 1, m is not required to be closed as the case for the collection of model space curves C_i, i = 1, n. When the collection is not closed, adding the appropriate sections of the boundary of the domain of the surface must make the collection closed. An example of an open collection of parameter space curves is that of a boundary going through a pole as shown in section 4.9.3.

4.9.3 Figure 11: Boundary

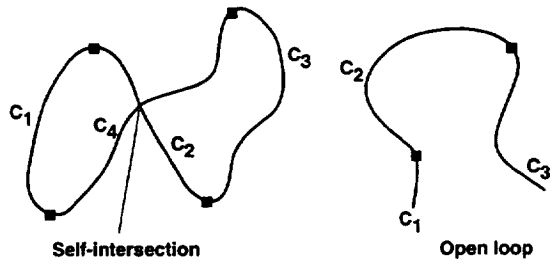


(a)

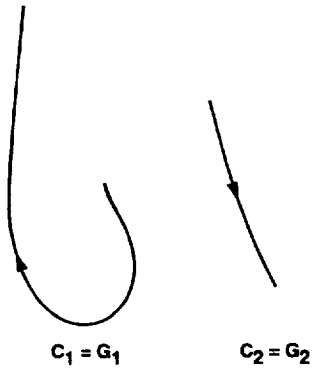
Figure 11.(a) Boundary: domain space and parameter space curves.



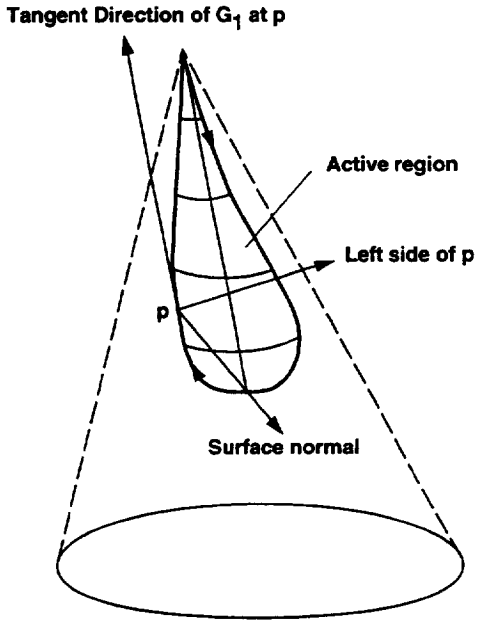
(b)



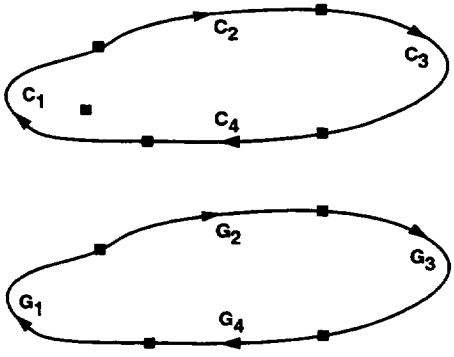
(e)



(c)

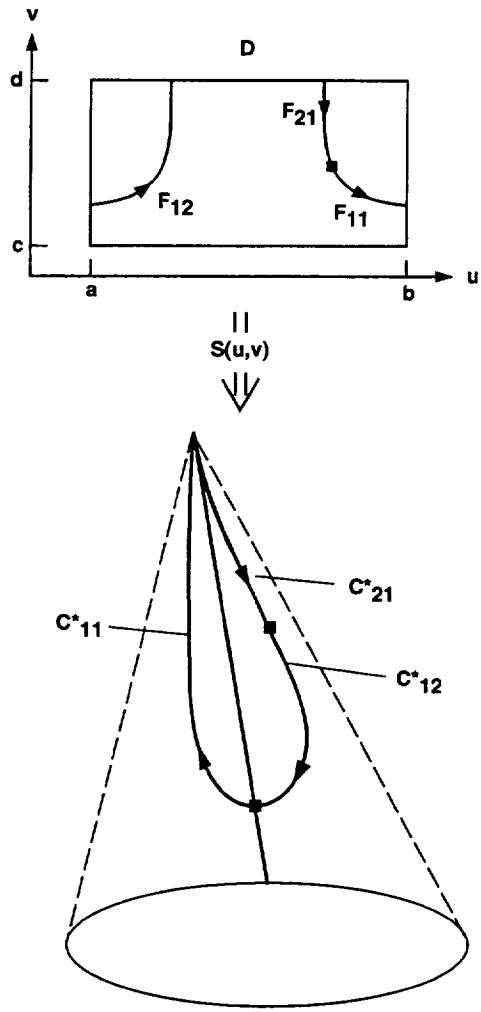


(f)



(d)

Figure 11. Continued. (b) Boundary: an allowed surface with model space curves, (c) boundary: model space curves from figure 11(b), (d) boundary: model space curves, C_i, of a boundary, and the corresponding oriented curves, G_i, (e) boundary: examples of disallowed model space curves, (f) boundary: active region defined by the boundary in figure 11(b),



(g)

Figure 11. Concluded. (g) Boundary: parameter space curves and their mapping onto the surface.

4.9.4 Boundary: Database Information

The database contains the following information for boundary entity.

trimType — A flag to indicate the preferred representation of the trimming curves in the sending system:

0 - Unspecified

1 - Model space

2 - Parameter space

3 - Representations are of equal preference

If trimType = 1, C_i are more accurate than C_i^*.

If trimType = 2, C_i^* are more accurate than C_i.

surface — The pointer to the untrimmed B-spline surface entity to be bounded

nCurves — The number of boundary curve objects (see below).

boundaryCurves — An array of boundary curve objects (see below). The curves are ordered according to the definitions in 4.9.2.

Each boundary curve object contains the following information:

msCurve — The model space curve

orientation — An orientation flag indicating whether the direction of the model space curve should be reversed before being used in the boundary. The possible choices for the flag are:

No reversal:

The direction of the model space curve does not require reversal, and

Requires reversal:

The direction of the model space curve must be reversed.

nPsCurves — The number of parameter space curves corresponding to this model space curve.

psCurves — The array of pointers to the corresponding parameter space curves. The curves are ordered to produce the same orientation as the model space curve.

4.9.5 Boundary: Note

The boundary entity is frequently used to specify the boundary of a bounded surface, hence a boundary entity is usually a part of a bounded surface entity.

4.10 Bounded Surface (IGES Entity Type Number 143)

4.10.1 Bounded Surface: General Description

A bounded surface is a trimmed surface, i.e., a surface with a portion cut off. The boundary of the bounded surface is defined by the boundary entity.

4.10.2 Bounded Surface: Mathematical Formulation

A bounded surface consists of an untrimmed surface and a number of boundary entities, which define the boundary of the bounded surface. The definitions and constraints on the boundary and the untrimmed surface are the same as those described in the boundary entity.

4.10.3 Figure 12: Bounded Surface

The bounded surface with the boundary in section 4.9.3 is shown below.

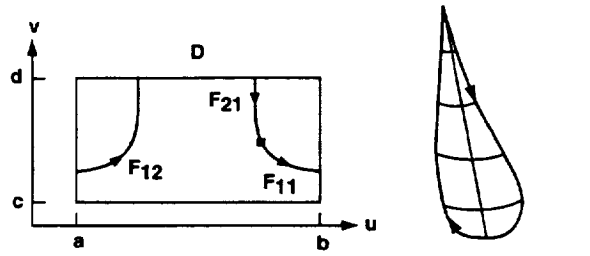


Figure 12. Bounded surface: domain space and parameter space curves and the bounded surface with the boundary in section 4.10.3.

4.10.4 Bounded Surface: Database Information

The database contains the following information about a bounded surface.

surface — The pointer to the untrimmed B-spline surface entity to be bounded.

nBound — The number of boundary entities

boundary — The pointers to boundary entities. The boundary entities are unordered.

4.11 Curve on a Parametric Surface (IGES Entity Type Number 142)

4.11.1 Curve on a Parametric Surface: General Description

The curve on a parametric surface entity associates a given curve with a surface and identifies the curve as lying on the surface.

4.11.2 Curve on a Parametric Surface: Mathematical Formulation

Let S(u,v) = (X(u,v), Y(u,v), Z(u,v)) be a regular parametrized surface whose domain is a rectangle defined by

$$D = \{(u,v) \mid u_1 <= u <= u_2 \text{ and } v_1 <= v <= v_2\}$$

Let G(t) be a curve defined by

$$G(t) = (u(t),v(t)) \text{ for } a <= t <= b$$

taking its values in D.

A curve C(t) on the surface S(u,v) is the composition of two mappings, S and G, defined as follows:

$$\begin{aligned} C(t) &= S \cdot G(t) \\ &= S(G(t)) \\ &= S(u(t),v(t)) \\ &= (x(u(t),v(t)),\ y(u(t),v(t)),z(u(t),v(t))) \end{aligned}$$

The curve G lies in the 2-D space which is the domain of the surface S. Therefore, the representation used for G which has been derived from a curve defined previously must be two-dimensional: the X and Y coordinates of this curve are used. A curve on a parametric surface is given by:

1. the mapping C and an indication that the curve lies on the surface S(u,v)

2. the mappings G and S whose composition gives the curve C.

4.11.3 Figure 13: Curve on a Parametric Surface

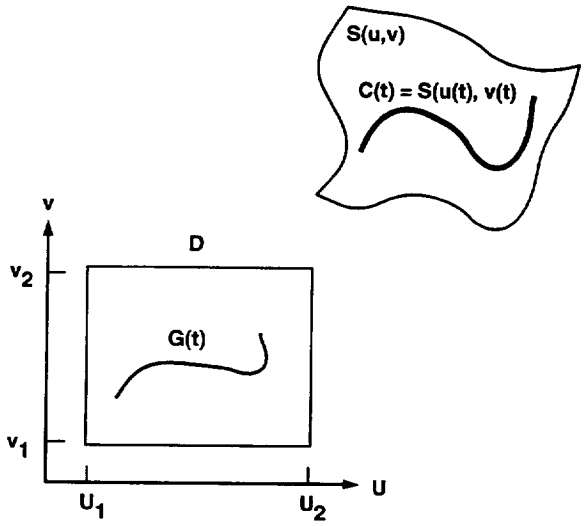


Figure 13. Curve on a parametric surface: domain space and parameter space curves.

4.11.4 Curve on a Parametric Surface: Database Information

The database contains the following information about a curve on the parametric surface.

createType — A flag to indicate the way the curve on the surface has been created. The possible values are:

- Unspecified
- Projection of a given curve on the surface
- Intersection of two surfaces
- Isoparametric curve, i.e., either a u-parametric or a v-parametric curve

surface — The pointer to the surface on which the curve lies

psCurve — The pointer to the parameter space curve, G

msCurve — The pointer to the model space curve, C

pref — A flag to indicate preferred representation of the sending system. Possible values are:

- Unspecified
- S · G is preferred
- C is preferred
- C and S · G are equally preferred.

When S · G is preferred, S · G is more accurate than C. When C is preferred, C is more accurate than S · G.

4.11.5 Curve on Parametric Surface: Note

In IGES, the curve on the parametric surface entity is represented in two ways. It can either be shown with the trimmed surface entity to form a trimmed surface or represented as a curve on a surface, such as a projected curve on a surface. These representations are replaced by the boundary entity in this specification. Only the latter use is allowed for the curve on a parametric surface in this specification. This entity allows an association between a curve and a surface. However, there are systems that do not keep this association, hence do not produce this entity for a curve even if the curve is on a surface.

5.0 Nongeometric Entity Description

5.1 Transformation Matrix (IGES Entity Type Number 124, Forms 0, 1)

5.1.1 Transformation Matrix: General Description

Most of the geometry in the database is specified in the definition space of the individual entities. The definition space of an entity is the most convenient coordinate system for describing the particular entity. The model space is where all the geometry components are assembled together. To position the entity into model space, the entity must be transformed from its definition space to the model space. This is done by applying one or more transformation matrices.

5.1.2 Transformation Matrix: Mathematical Formulation

The transformation matrix transforms three-row column vectors by means of a matrix multiplication and then a vector addition. The notation for this transformation is:

$$\begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \begin{bmatrix} X_{input} \\ Y_{input} \\ Z_{input} \end{bmatrix} + \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix} = \begin{bmatrix} X_{output} \\ Y_{output} \\ Z_{output} \end{bmatrix}$$

The column vector $[X_{input}, Y_{input}, Z_{input}]$ is the vector being transformed and the column vector $[X_{output}, Y_{output}, Z_{output}]$ lists the results of the transformation. The input vectors are points in the original space. The output vectors are points in the new space. All coordinate systems are assumed to be Cartesian and right-handed.

Coordinate systems may be related successively to each other. For example, if the coordinate system change involving the matrix R_2 and the translation vector T_2 is to be applied following the coordinate system change involving the matrix R_1 and the translation vector T_1, then the matrix R and the translation vector T expressing the combined changes are

$$R = (R_2)(R_1) \text{ and } T = (R_2)(T_1) + T_2$$

Successive coordinate system changes are specified by allowing a transformation matrix entity to reference another transformation matrix entity through the transformation matrix pointer. In the example above, the transformation matrix entity containing R_1 and T_1 would contain a transformation matrix pointer to a transformation matrix entity containing R_2 and T_2. The general rule is that transformation matrix entities applied earlier in a succession will reference transformation matrix entities applied later.

A 3×3 matrix R is called orthogonal provided its transpose, R^t, yields a matrix inverse for R. The columns of an orthogonal matrix considered as vectors form an orthogonal collection of unit vectors. As $(R^t)^t = R$, the transpose of an orthogonal matrix is again an orthogonal matrix. The determinant of an orthogonal matrix is equal to either plus one or minus one. In the event that R is an orthogonal matrix with the determinant equal to positive one, R can be expressed as a rotation about an axis passing through the origin. In this event, R is referred to as a rotation matrix. In the event that R is an orthogonal matrix with the determinant equal to negative one, R can be expressed as a rotation about an axis passing through the origin followed by a reflection about a plane passing through the origin perpendicular to the axis of rotation.

There are two forms of the transformation matrix entity.

Form 0: (The default form)

R is an orthogonal matrix with the determinant equal to H. T is arbitrary. The columns of R taken in order form a right-handed triple in the output coordinate system.

Form 1:

R is an orthogonal matrix with the determinant equal to negative one. T is arbitrary.

5.1.3 Transformation Matrix: Database Information

The database contains the following information for a transformation matrix.

tmat The translation vector $[T_1\ T_2\ T_3]$

rmat The matrix $\begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix}$

5.2 General Note (IGES Entity Type Number 212)

5.2.1 General Note: General Description

A general note entity consists of one or more text strings and is used to pass on textual information about the geometry. This can include information on the history of

the object, reference documents, textual descriptions, etc. A general note entity can exist separately or can be associated with some entity or entities.

5.2.2 General Note: Definition

The text font is the ASCII Character Set for a general note. Each text string can have width, height, slant angle and rotation angle specified (see figures in section 5.2.3). Each text string lies on a constant Z-plane in the definition space of the general note. Transformation matrices are used to bring the general note to the desired location in the model space.

Within the definition space, the parameters for a text box containing a text string are applied in the following order:

1. Define the box height (boxHeight) and box width (boxWidth).

The rotateInternalText flag, see 5.2.4, indicates whether the text box is filled with horizontal text or vertical text. The box width is measured from the start of the left-most (first) text character in the positive X direction along the text base line, and extends to the end of the right-most (last) character, extending nCharacters, see 5.2.4, and (nCharacters - 1) intercharacter spaces. The box height is measured in the positive Y direction and is the height of capital letters. It is equivalent to the symbol "h" used in appendix C of reference 4. The box height and width are measured before the rotation angle (see no. 3) is applied. The text start point is defined as the lower left corner of the first character box.

2. The slant angle is then applied to each individual character. For horizontal text, it is measured from the X axis in a counterclockwise direction. For vertical text, the slant angle is measured from the Y axis.

3. The rotation angle is then applied to the text block. This rotation is applied in a counterclockwise direction about the text start point. The plane of rotation is the X, Y plane at the depth Z, where Z is the value given for the text start point.

4. The mirror operation is performed next. mirrorFlag = MirrorPerpBase, see 5.2.4, indicates the mirror axis is the line perpendicular to the text base line and through the text start point. mirrorFlag = MirrorBase indicates the mirror axis is the text base line.

Finally, the transformation matrix entity is used to specify the relative position of the definition space within the model space.

5.2.3 Figure 14: General Note

The following figure shows the parameters defining a text string. The text string "VERTICAL TEXT" has rotateInternalText set equal toVerticalText.

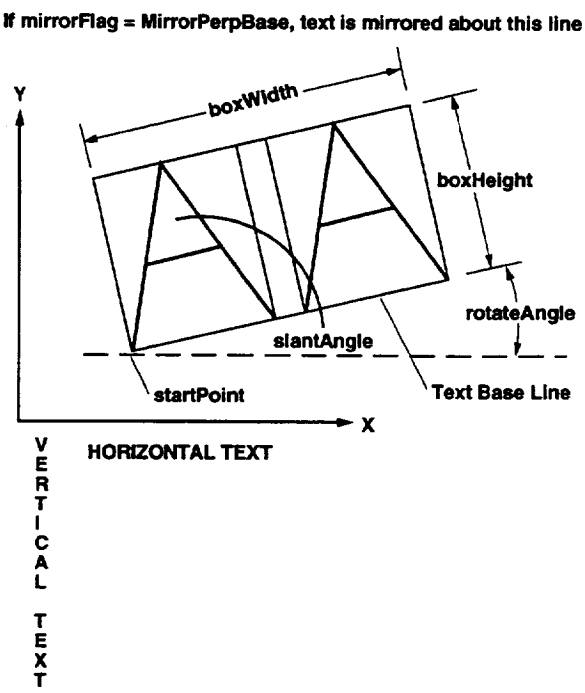


Figure 14. General note.

5.2.4 General Note: Database Information

The database contains the following information for a general note.

nStrings	Number of text string objects in a general note (see below).
strings	An array of text string objects (see below).

Each text string object contains the following information:

nCharacters	Number of characters in the text string. nCharacters must always be equal to the character count of the text string.
boxWidth	The width of the box contains the text string.

boxHeight	The height of the box contains the text string. The box is the height of the characters in the text string.
slantAngle	Slant angle of the text string in radians. ($\pi/2$ is the value for no slant and is the default value.)
rotateAngle	Rotation angle of the text string in radians.
mirrorFlag	Mirror flag: NoMirror - no mirroring MirrorPerpBase - mirror axis is perpendicular to text base line MirrorBase - mirror axis is text base line
rotateInternalText	The rotate internal text flag: HorizontalText- horizontal text VerticalText - vertical text
startPoint	Text start point, with X, Y, and Z components.
string	The text string

5.3 Subfigure Definition (IGES Entity Type Number 308)

5.3.1 Subfigure Definition: General Description

The subfigure definition and subfigure instance entities allow one copy of the geometry to be placed in many locations in a design without duplicating the geometry. For example, in a turbine engine design all the turbine blades on the same stage are identical in shape. If the geometry for all the blades is stored the database may become very large. Alternatively, a generic turbine blade geometry is defined with the subfigure definition entity. All the blades are instances of this generic blade with location transformation.

A subfigure definition can contain other subfigure instances. That is, subfigures can be nested. The field depth in the database indicates the level of nesting of the subfigure. If depth = 0, the subfigure has no references to any subfigure instances. A subfigure can not reference a subfigure instance that has equal or greater depth. A depth = N indicates there is a reference to an instance of a subfigure defintion with depth = N-1.

5.3.2 Subfigure Definition: Database Information

The database contains the following information for a subfigure definition.

depth	Depth of the subfigure (indicates the amount of nesting).
name	Subfigure name.
nEntity	Number of entities in the subfigure definition.
entities	The pointers to the entities that define the subfigure.

5.4 Color Definition (IGES Entity Type Number 314)

5.4.1 Color Definition: General Description

Color definition is used to define colors for displaying geometries. A color is defined by its red, green, and blue component values. Each geometry can be assigned with one color.

5.4.2 Color Definition: Database Information

The database contains the following information about a color definition.

R	Red color component as a percent of full intensity
G	Green color component as a percent of full intensity
B	Blue color component as a percent of full intensity
name	The optional color name. It is a character string which may contain some verbal description of the color.

5.5. Group Associativity (IGES Entity Type Number 402, Forms 1, 7, 14, and 15)

5.5.1 Group Associativity: General Description

The group associativity allows a collection (of a set) of entities to be maintained as a single, logical entity. A group is to represent a logical concept by relating entities of a common property together, for example, all points can be grouped to form a point group.

A group can also be used to collect entities logically to form a meaningful geometric class, for example, all the cross section curves of a wing are grouped together to form a cross sectional presentation of a wing.

There are four forms (Forms 1, 7, 14, and 15) in this entity type. Forms 1 and 7 indicate an unordered group, a group whose constituent entities form an unordered set. Forms 14 and 15 indicate an ordered group, a group whose constituent entities are in the same order as they are listed. Entities in a group with Forms 1 or 14 have back pointers to the group entity. Entities in a group with Forms 7 and 15 do not have back pointers to the group entity. With the back pointer, given an entity in a group, it is possible to follow the back pointer to locate its group entity without searching through the entire database.

5.5.2 Group Associativity: Database Information

The database contains the following information for a group associativity.

nEntities	Number of entities in the group
entities	The pointers to the entities. Depending on the form number, the pointers are either ordered or unordered.

5.6 Name (IGES Entity Type Number 406, Form 15)

5.6.1 Name: General Description

"Name" is a text string to be associated with an entity or a group of entities (formed by group associativity). This entity is used to give a meaningful identification to a particular piece of the geometry. For example, the geometry of a wing is assigned the name "wing." Longer comments should be assigned to the geometry via general notes.

5.6.2 Name: Database Information

The database contains the following information for a name.

string	The text string which is the name.

5.7 Singular Subfigure Instance (IGESEntity Type Number 408)

5.7.1 Singular Subfigure Instance: General Description

The subfigure definition and subfigure instance entities allow one copy of the geometry to be placed in many locations in a design without duplicating the geometry. The singular subfigure instance entity represents a single instance of a defined subfigure. See section 5.3 for more information. The subfigure can be scaled and translated to its new position based on data in the database. It can also be translated and rotated by the transformation matrix associated with the singular subfigure instance entity.

5.7.2 Singular Subfigure Instance: Database Information

The database contains the following information for the instance of a singular subfigure instance.

pointer	The pointer to the subfigure definition entity.
X,Y,Z	The translation vector to place the instance.
scale	The scale factor for the instance.

6.0 Database Details

Since we expect most of the users of this database information format to import the geometry in the database from IGES files, the database is organized closely following the IGES file format. By doing so, information mapping from an IGES file to the database can be accomplished more easily and a minimum amount of information will be lost.

As mentioned in section 3, the database information that varies from entity to entity is described in sections 4 and 5. Section 6.1 describes the information that is common to all the entities (corresponding to IGES Directory Entry Section). Section 6.2 describes the global data information (corresponding to IGES Global Section). Section 6.3 describes the mechanism through which the information in the database is accessed by applications.

All the database descriptions are at a high level and hence do not specify the exact data stored, individual data types and data structures, or the exact storage formats.

6.1. Entity-Independent Information

The database contains the following entity-independent information.

Entity Type — Identify the type of the entity

Line Font Pattern — The display pattern to be used when displaying the entity in a line drawing mode.

Possible values are:

- No pattern specified
- Solid
- Dashed
- Phantom
- Centerline
- Dotted

Level — The display level on which the entity resides. An entity can reside in only one level.

Transformation Matrix Pointer — These data are either zero for the identity rotation matrix and zero translation vector or a pointer to a transformation matrix entity, which is used in transforming the entity from the definition space to model space. Also see section 3.2.

Line Weight Number — The display thickness of the lines when the entity is displayed in line drawing mode. See section 6.1.1 for more information.

Color — Either a color token (see below) or a negated pointer to a color definition entity (section 5.3). The color tokens are:

- No color assigned
- Black
- Red
- Green
- Blue
- Yellow
- Magenta
- Cyan
- White

Form — Certain entities have different interpretations. These interpretations are uniquely identified by a form. Possible form values are listed within each entity description.

Entity Label — Up to eight alphanumeric characters.

Used as the label of an entity.

Entity Subscript — Number 1- to 8-digit unsigned number associated with entity label as its subscript.

Status — Status of the entity (see section 6.1.2)

Additional Pointers

Additional pointers are either back pointers with group associativity entities and/or pointers to properties as described in section 6.1.3.

6.1.1 Line Weight Number

This value denotes the thickness (or width) with which an entity should be displayed in a line-drawing mode. A series of possible thicknesses is specified by data in the global information (section 6.2). The largest thickness

possible is that specified by the global parameter, width of maximum line weight in units. The smallest thickness possible is equal to the result of dividing the width of maximum line weight in units by maximum number of line weight gradations (also a global parameter) and is denoted by setting the line weight number equal to 1. The thickness between the smallest and largest thickness are available in increments equal to the smallest possible thickness and are denoted by setting the Line Weight Number equal to the integer number of adjacent increments required.

Thus, display thickness is:

line_weight_number * (width_of_maximum_line_weight_in units/maximum_number_of_line_weight_gradations).

A value of 0 indicates that the default line weight display of the system is to be used.

6.1.2 Status

Status encodes three items: blank status, subordinate entity switch status, and hierarchy.

6.1.2.1 Blank Status

Blank status defines whether the entity is visible on the display.

Possible values are visible and blanked.

6.1.2.2 Subordinate Entity Switch

This switch indicates whether the entity is referenced by other entities in the database and, if so, what type of relationship exists. An entity can be independent, physically dependent, logically dependent, or both physically and logically dependent. The possible values are defined as follows:

Independent – The entity is not referenced by any other entities in the database. It can exist alone in the database.

Physically Dependent – This entity (the child) is referenced by another entity (the parent) in the database. The child cannot exist unless the parent exists. The transformation matrix pointed to by the entity (as a child) must be applied to the entity's definition in order to determine its location in the parent's definition space.

Entity A is dependent on entity B if, and only if, the parameter data entry of entity B contains a pointer to entity A. The additional pointers as defined in section 6.1.3 are ignored for the purposes of this definition.

The structure formed by a parent entity and its physically dependent components is indivisible and may therefore be considered to form a single entity. An example of a physically dependent entity is that of a circular arc entity pointed to by a composite curve entity.

Logically dependent – This entity (the child) can exist alone in the database, but is referenced by some sort of logical grouping entity, or entities (the parents), such as group associativities. The transformation matrix pointed to by the parent entity has no effect on the location of the child.

An example of a logically dependent entity is that of a line entity pointed to by a group associativity entity.

Both physically and logically dependent – This entity is physically dependent upon another entity in the file and is subject to the physical dependency rules described above. This entity is also referenced by one or more logical grouping entities, and is also subject to the logical dependency rules described above. Additionally, an entity cannot be physically and logically dependent upon the same parent entity. The case of an entity being both logically and physically dependent refers to the fact that the transformation matrix pointed to by a parent entity is not applied to its logically dependent children.

An example of a logically and physically dependent entity is that of a line entity occurring in a composite curve entity and pointed to by a group associativity entity.

6.1.2.3 Hierarchy

This flag indicates the relationship between entities in a hierarchical structure and is used to determine which entity's display attributes shall apply. It controls the following display attributes: line font, entity level, blank status, line weight, and color. Possible values are "Apply to dependent" and "Do not apply to dependent." The possible values are defined as follows:

Apply to dependent

All the display attributes of this entity apply to entities physically dependent on to this entity. Consequently, the attributes assigned to all the subordinate entities are ignored.

Do not apply to dependent

None of the display attributes of this entity applies to entities physically dependent on this entity. Consequently, the attributes assigned to this entity are ignored.

6.1.3 Additional Pointers

Each entity may contain additional pointers. Each of the entities in a group associativity entity with Form 1 or 14 has a back pointer to the group associativity entity. This back pointer is one of the additional pointers.

Entities with a general note entity as an annotation or a name entity as a property contain pointers to the entity of the annotation or property. These pointers are also the additional pointers.

6.2 Global Section Information

The database contains the following global information:

Product Identification

This is a text string which is the name or identifier for the model.

Model Space Scale

This is the ratio of model space to real world space. To obtain the real world size of the model, the model in the database shall be divided by this scale.

Unit Flag

The flag that indicates the measuring system used in the database. The available units are:

- Inches
- Millimeters
- Feet
- Miles
- Meters
- Kilometers
- Mils (i.e., 0.001 inch)
- Microns
- Centimeters
- Microinches

Maximum Number of Line Weight Gradations

This is the number of equal subdivisions of line thickness. The default value is 1.

Width of Maximum Line Weight in Units

This is the actual width of the thickest line possible in the database in database units.

Date and Time of Exchange File Generation

If the database is originally loaded from an IGES file, this is the time stamp indicating when the file was generated.

Minimum User-Intended Resolution

This parameter indicates the smallest distance in model space units that the system should consider as discernible. Coordinate locations in the database which are less than this distance apart should be considered to be coincident.

Approximate Maximum Coordinate Value

This is the upper bound on the values of all coordinate data actually occurring in this model expressed in the units of the database. The absolute magnitude of each of the coordinates in this model is less than or equal to this value.

Name of Author

If the database is originally loaded from an IGES file, the author is the person responsible for the creation of the data contained in the IGES file.

Author's Organization

If the database is originally loaded from an IGES file, this is the organization or group with whom the author of the IGES file is associated.

Version Number

If the database is originally loaded from an IGES file, this is a value that denotes the version of the IGES Specification used to create the IGES file. Since this number depends on the version of the IGES Specification, please refer to the IGES Specification for the mapping of the value to the true IGES version.

Date and Time Model Was Created or Modified

If the database is originally loaded from an IGES file, this is a time stamp indicating when the model was created or last modified, whichever occurred last.

6.3 Database Handle

The database provides two handles through which the applications can access the data in the database. One handle is a structure through which the information in section 6.2 can be accessed. The other handle is a structure through which all the entities in the database can be accessed. This specification does not impose any organizational constraints to the structures. Some possible implementations for the first structure are an array, list, table, or object. Some possible implementations for the second structure are hash tables, unordered lists, and organized lists by entity types, etc.

Ames Research Center
National Aeronautics and Space Administration
Moffett Field, California 94035-1000
September 8, 1993

References

1. The Initial Graphics Exchange Specification (IGES), distributed by National Computer Graphics Association, Administrator, IGES/PDES Organization, 2722 Merrilee Drive, Suite 200, Fairfax, Va.

2. Thomas, G.: Calculus and Analytic Geometry, Addison-Wesley, 1960.

3. Farin, G.: Curves and Surfaces for Computer Aided Geometric Design, Academic Press, 1988.

4. Dimensioning and Tolerancing, (Y14.5M-1982), American National Standard Institute, 1982.

5. The National Institute of Standards and Technology (NIST), U. S. Department of Commerce, Gaithersburg, Md.

Bibliography

Bartels, R. H.; Beatty, J. C.; and Barsky, B. A.: An Introduction to Splines for Use in Computer Graphics and Geometric Modeling, Morgan-Kaufmann, Palo Alto, Calif., 1987.

Boehm, W.; Farin, G.; and Kahmann, J.: A Survey of Curve and Surface Methods in CAGD. Computer Aided Geometry Design, vol. 1, no. 1, July 1984, pp. 1–60.

de Boor, C.: A Practical Guide to Splines, Springer-Verlag, New York, 1978.

Farin, G.: Curves and Surfaces for Computer Aided Geometric Design, A Practical Guide, Academic Press Inc., 1990.

Lee, E.T.Y.: Rational Quadratic Bezier Representation for Conics, in Geometric Modeling: Algorithms and New Trends, G. Farin, ed., SIAM Philadelphia, 1987, pp. 3–19.

Tiller, W.: Rational B-Splines for Curve and Surface Representation, CG&A, vol. 3, no. 10, Sept. 1983, pp. 61–69.

Piegland, L.; and Tiller, W.: Curve and Surface Constructions Using Rational B-Splines, Computer-Aided Design, vol. 19, no. 9, Nov. 1987, pp. 485–498.

Piegland, L.; and Tiller, W.: A Menagerie of Rational B-Spline Circles. IEEE Computer Graphics and Applications, vol. 9, no. 5, Sept. 1989, pp. 48–56.

Appendix A

Specific IGES Protocol for Geometry Data Transfer for Computational Fluid Dynamics

NASA-IGES and NASA-IGES-NURBS-Only

TABLE OF CONTENTS

1.0 General

This specification provides a detailed description of the specific IGES Protocol for geometry data transfer for computational fluid dynamics, referred to as NASA-IGES. A more restrictive (and highly recommended) IGES Protocol is also specified, refered to as NASA-IGES-NURBS-Only.

This specification does not provide all the details necessary to utilize IGES files. The developer of software for reading or writing any IGES, files will need to review the IGES document in reference 1 for a complete description of the file format.

This specification does provide a listing of the entities and restrictions placed on NASA-IGES and NASA-IGES-NURBS-Only files. NASA-IGES is a subset of standard IGES and NASA-IGES-NURBS-Only is a subset of NASA-IGES. All NASA-IGES-NURBS-Only and NASA-IGES files are valid IGES files.

Throughout this document reference is made to NASA-IGES. These comments also pertain to NASA-IGES-NURBS-Only files. All comments and restrictions are the same for both types of files except as noted. NASA-IGES files can include seven additional entities not allowed in NASA-IGES-NURBS-Only files. These are identified in sections 3.3 and 3.4.

Several general recommendations and entity conversion specifications are provided for entities not allowed under this Specification. For a more detailed discussion on why certain IGES entities were included in or excluded from this specification, please see the Geometry Data Exchange Subcommittee Notes dated September 30, 1991 and October 1, 1992. To obtain a copy, contact Matthew Blake, MS T27B-2, NASA Ames Research Center, Moffett Field, CA 94035-1000.

This document is organized as follows:

Section 1.0 of this Specification contains general information.

Section 2.0 contains recommended practices that will provide smoother data transfer.

Section 3.0 contains all specifications for a NASA-IGES or NASA-IGES-NURBS-Only data file. This includes specifications of the behavior of data file preprocessors and postprocessors.

2.0 Comments and Recommendations

2.1 Bidirectional Requirement

Use of this specification is intended to be bidirectional. This means software systems should be capable of both reading and writing data in NASA-IGES format. This will require CAD systems to not only write data in NASA-IGES format but also to read in a modified geometry configuration and merge the data into the original database. Grid generation programs will be required to both read in NASA-IGES data and to write out NASA-IGES files if the geometry is modified.

2.2 Recommendations

To the designers of CAD systems and grid generation software, it is with the strongest of recommendations that we state:

Please follow the recommendations in section 3.0

This will help to ensure that your software can utilize the geometry data represented in NASA-IGES format to the fullest extent. This is extremely important as many NASA grid generation and analysis packages will be modified to utilize only data represented by the recommended methods. An example of this is the clear recommendation to utilize NURBS geometry (Entity Type 126) over Conic (Entity Type 104) representations.

Utilize the higher order representation methods to pass geometry data. Do not convert higher order data to Copious Data (Entity Type 106) or any other point format—that would not follow the intent of this specification.

If possible, utilize the NASA-IGES-NURBS-Only protocol rather than just the NASA-IGES protocol. This will avoid the need for translation of some of the geometry entities prior to use by the grid generation software.

3.0 Format Specification

3.1 IGES Compatibility

This specification is a subset of The Initial Graphics Exchange Specification (IGES) Version 5.1 (ref. 1), National Institute of Standards and Technology (NIST) number NISTIR 4412 (ref. 6).

There are no items in this specification that do not adhere to the standard IGES format. There are no IGES entities requiring or utilizing any implementor defined types.

There are five classes of IGES entities:

1. Curve and surface geometry entities
2. Constructive solid geometry (CSG) entities
3. B-Rep solid entities
4. Annotation entities
5. Structure entities

The NASA-IGES protocol includes a total of 19 entities (see sections 3.3 and 3.4) consisting of 12 geometry entities, no CSG entities, 1 annotation entity, and 6 structure entities.

The NASA-IGES-NURBS-Only protocol includes a total of 12 entities (see sections 3.3 and 3.4) consisting of 7 geometry entities, no CSG entities, 1 annotation entity, and 4 structure entities.

The entities included in this specification are referred to as NASA-IGES entities. Non-NASA-IGES entities are those IGES entities not included in this specification. A NASA-IGES file is a valid IGES file that contains only NASA-IGES entities.

3.2 Behavior of Conforming IGES File Transfer Programs

A *preprocessor* produces an IGES file from the internal database of a system. A preprocessor may need to translate the internal representations of the system to the available IGES representations. Usually the preprocessor is attached to a CAD system and is writing a file to be used by a grid generation or analysis package.

A *postprocessor* reads an IGES file into the internal database of a system. A postprocessor may need to translate the entities in the file into the internal formats used in the system. Usually the postprocessor is attached to a grid generator and is reading a file generated by a CAD system.

A *translator* translates IGES files from one flavor into another, e.g., from the IGES files from a particular CAD system to IGES files adhering to this specification.

Since the data file defined by this specification is an IGES file, conforming preprocessors or postprocessors must obey all the conformance rules in IGES (see section 1.3, IGES V5.1). Additional conformance rules for this Specification are discussed in the following subsections.

It is strongly recommended that systems which generate geometry data for CFD grid generation produce NASA-IGES-NURBS-Only files. This will ensure the highest acceptability of the data files by the grid generation and analysis software, since most of this software will only be equipped with a NASA-IGES restrictive reader (section 3.2.1.1) and the availability of translators (section 3.2.3) is uncertain.

3.2.1 Postprocessors (Readers)

This specification defines two classes of NASA-IGES conforming readers. These two classes are restrictive readers and non-restrictive readers. None of the readers translate non-NASA-IGES entities.

In practice, a translator and a reader can be combined into one unit. Such a postprocessor will read in and translate non-NASA-IGES entities.

3.2.1.1 Restrictive Readers

A NASA-IGES restrictive reader shall be capable of reading any NASA-IGES file. The entities can only have pointers to other NASA-IGES entities. When a restrictive reader encounters a non-NASA-IGES entity, the reader shall notify the user of the encounter. Following this notification, additional reader behavior is undefined.

3.2.1.2 Non-Restrictive Readers

A NASA-IGES non-restrictive reader conforming to this specification shall not read in non-NASA-IGES entities; however, the existence of the non-NASA-IGES entities should not stop the reading process. Entities with a pointer to a non-NASA-IGES entity should have the pointer ignored (i.e., the entity pointed to by the pointer should

not be read in) and if 0 is an allowed option for the pointer field, the field value should be replaced by 0. If ignoring the pointer invalidates the geometry of the current entity, such as might happen for a composite curve entity, the current entity should not be read in.

Whenever it is necessary for the reader to ignore a pointer, the rules in section 2.2.4.3.9.2 of IGES Version 5.1 regarding subordinate entity switch also apply. The rule can be stated as: if the parent entity of a child entity with a physically dependent switch is not read in, the child entity cannot exist.

For example, the component curves of a composite curve entity should have their subordinate entity switches set as physically dependent (page 56, IGES version 5.1) (ref. 1). According to the above rules, if one of the component curves is a non-NASA-IGES entity, the composite curve entity and all its component curves should not be read in.

3.2.1.3 Postprocessor Data Utilization

Any program reading in IGES data defined in this specification is NASA-IGES Read Utilization Conforming if it utilizes all geometry data present in a NASA-IGES file in the program's internal database.

Geometry entities which are not intended to define the geometry (for drafting, etc.) are not required to be utilized by the reading program.

3.2.2 Preprocessors (Writers)

This specification defines two classes of NASA-IGES conforming writers. These two classes are restrictive writers and non-restrictive writers.

In practice, a translator and a writer can be combined into one unit. Such a preprocessor will translate the internal data to NASA-IGES entities and then write out the entities to an IGES file.

3.2.2.1 Restrictive Writer

A NASA-IGES restrictive writer shall create IGES files containing only NASA-IGES entities. The entities can only have pointers to other NASA-IGES entities.

3.2.2.2 Non-Restrictive Writer

A NASA-IGES non-restrictive writer can generate any valid IGES entity. This is a standard IGES conforming preprocessor and does not have any NASA-IGES specific capabilities or restrictions.

3.2.2.3 Preprocessor Data Utilization

Any program writing out NASA-IGES data is NASA-IGES write utilization conforming if the system writes out all the geometry into the IGES file. Any non-geometry related geometric entities (for drafting, etc.) can be ignored and not written to the file.

3.2.3 Translators

A conforming IGES translator shall convert an IGES file to a NASA-IGES file. The following provides the mapping for those entities that should be converted by the translator. Treatment of other non-NASA-IGES entities by the translator is defined in section 3.2.3.2.

3.2.3.1 Entity Conversion Maps

3.2.3.1.1 NASA-IGES Conversion Map

A conforming NASA-IGES translator shall convert an entity on the left column of the map to the corresponding entity on the right in table 1.

3.2.3.1.2 NASA-IGES-NURBS-Only Conversion Map

In addition to the NASA-IGES conversions specified in 3.2.3.1.1, a conforming NASA-IGES-NURBS-Only translator shall convert an entity on the left column of this map to the corresponding entity on the right in table 2.

3.2.3.1.3 Conversion Procedures

When a mathematically exact conversion exists, the exact conversion, instead of an approximation, should be used. For example, the uniform offset curve entity (Entity Type 126, Flag 1) of a circular arc (Entity Type 100) should be a circular arc (Entity Type 100). The uniform and linear offset curve entity (Entity Type 126, Flags 1 and 2) of a line (Entity Type 110) should be a line (Entity Type 110). The ruled surface (Entity Type 118, Form 1) formed between two B-spline railing curves (Entity Type 126) should be converted exactly to a B-spline surface (Entity Type 128).

When no exact conversions exist, it is preferred for the translator to provide a method for the user to control the accuracy of the conversion so that the user can change it if necessary.

The Form field in Entity Type 126 and 128 should be set to an appropriate value other than 0 if the curve or surface is in one of the available forms in IGES. For example, the

Table 1: NASA-IGES Conversion Map

>From Entity Type (,Form)	>To Entity Type (,Form)
Copious Data, Type 106, Form 11	Rational B-Spline Curve Type 126, Form 0, Degree 1, PROP1 1, PROP3 1, PROP4 0
Copious Data, Type 106, Form 12	Rational B-Spline Curve Type 126, Form 0, Degree 1, PROP3 1, PROP4 0
Copious Data, Type 106, Form 13	Rational B-Spline Curve Type 126, Form 0, Degree 1, PROP3 1, PROP4 0, the information about the vectors associated with the points will be lost
Copious Data, Type 106, Form 63	Rational B-Spline Curve Type 126, Form 0, Degree 1, PROP1 1, PROP2 1, PROP3 1, PROP4 0
Parametric Spline Curve, Type 112	Rational B-Spline Curve, Type 126
Parametric Spline Surface, Type 114	Rational B-Spline Surface, Type 128
Ruled Surface, Type 118	Rational B-Spline Surface, Type 128
Surface of Revolution, Type 120	Rational B-Spline Surface, Type 128
Tabulated Cylinder, Type 122	Rational B-Spline Surface, Type 128
Offset Curve, Type 130	Rational B-Spline Curve, Type 126, Circular Arc Type 100 or Line Type 110 on exact conversion.
Offset Surface, Type 140	Rational B-Spline Surface, Type 128
Trimmed Parametric Surface, Type 144	Bounded Surface, Type 143
Definition Levels, Type 406, Form 1	The entity with this property is placed in the first level identified by this Definition Levels entity.

Table 2: NASA-IGES-NURBS-Only Conversion Map

>From Entity Type (,Form)	>To Entity Type (,Form)
Circular Arc, Type 100	Rational B-Spline Curve, Type 126, Form 2, Degree 1,PROP1 1, PROP3 0, PROP4 0
Composite Curve, Type 102	Rational B-Spline Curve, Type 126, Forms 1, 2, 3, 4, or 5 as appropriate, Degree 1, PROP1 1, PROP3 1, PROP4 0
Conic Arc, Type 104	Rational B-Spline Curve, Type 126, Forms 3, 4, or 5 as appropriate, Degree 1, PROP1 1, PROP4 0
Copious Data, Type 106, Form 1	Rational B-Spline Curve, Type 126, Form 0, Degree 1, PROP1 1, PROP3 1, PROP4 0
Copious Data, Type 106, Forms 2 or 3	Rational B-Spline Curve, Type 126, Form 0, Degree 1 PROP3 1, PROP4 0
Line, Type 110	Rational B-Spline Curve, Type 126, Form 1, Degree 1, PROP1 1, PROP3 1, PROP4 0
Singular Subfigure, Instance Type 408	A copy of the geometry using original entities. These entities are then converted as specified in these Conversion Maps

offset surface (Entity Type 140) of a sphere (Entity Type 128, Form 4) should be a sphere (Entity Type 128, Form 4).

3.2.3.2 Treatment of Other Entities

Non-NASA-IGES entities which are not in the above map should be ignored and should not appear in the output file. Entities which are physically dependent on the ignored entities should be ignored and should not appear in the output file (refer to section 3.2.1.2), even if they are NASA-IGES entities.

For example, in the Parameter Data section of the linear dimension entity (Entity Type 216), there is a pointer to the general note entity. Since the general note entity is

physically dependent on the linear dimension entity, the general note entity should not appear on the output file.

3.3 Summary of Entity Types

This section contains a table summarizing the entity types utilized by this specification and several tables grouping the entities by function.

3.3.1 Summary of All Allowed Entities

Table 3 contains a summary list, ordered by IGES Entity Type number, of all the entities allowed in NASA-IGES and NASA-IGES-NURBS-Only data files.

3.3.2 Summary of Entities by Usage

Tables 4-6 contain summary groupings of the entities by recommended usage.

Table 3: Summary of NASA-IGES Entities

IGES Entity No.	Entity Name	NASA-IGES	NURBS-Only
Entity 0	Null entity	yes	yes
Entity 100	Circular arc	yes	no
Entity 102	Composite curve	yes	no
Entity 104	Conic arc	yes	no
Entity 106	Copious data	yes	no
Entity 110	Line	yes	no
Entity 116	Point	yes	no
Entity 124	Transformation matrix	yes	yes
Entity 126	Rational B-spline curve	yes	yes
Entity 128	Rational B-spline surface	yes	yes
Entity 141	Boundary	yes	yes
Entity 142	Curve on a parametric surface	yes	yes
Entity 143	Bounded surface	yes	yes
Entity 212	General note	yes	yes
Entity 308	Subfigure definition	yes	no
Entity 314	Color definition	yes	yes
Entity 402	Associativity instance	yes	yes
Entity 406	Property, Form 15: name	yes	yes
Entity 408	Singular subfigure instance	yes	no

Table 4: Geometric Entities Allowed in NASA-IGES-NURBS-Only Files

IGES Entity No.	Entity Name	Entity Class (see [IGES])
Entity 124	Transformation matrix	Geometry
Entity 126	Rational B-spline curve	Geometry
Entity 128	Rational B-spline surface	Geometry
Entity 141	Boundary	Geometry
Entity 142	Curve on a parametric surface	Geometry
Entity 143	Bounded surface	Geometry

Table 5: NASA-IGES Entities Not Allowed in NASA-IGES-NURBS-Only Files

IGES Entity No.	Entity Name	Entity Class (see (ref. 1))
Entity 100	Circular arc	Geometry
Entity 102	Composite curve	Geomety
Entity 104	Conic arc	Geometry
Entity 106	Copious data	Geometry
Entity 110	Line	Geometry
Entity 116	Point	Geometry
Entity 308	Subfigure definition	Structure
Entity 408	Singular subfigure instance	Structure

Table 6: Non-Geometric Entities Allowed in NASA-IGES and NASA-IGES-NURBS-Only Files

IGES Entity No.	Entity Name	Entity Class (see (ref. 1))
Entity 0	Null entity	Structure
Entity 212	General note	Annotation
Entity 314	Color definition	Structure
Entity 402	Associativity instance	Structure
Entity 406	Property, Form 15: name	Structure

It is desirable to represent all geometric objects utilizing the following entities which are available in NASA-IGES and NASA-IGES-NURBS-Only files:

3.4 Specific Entity Types

This section contains a detailed explanation of each IGES entity utilized by this specification.

Each entity section has three subsections covering the following

1. Usage: Explain the general usage and how to use any options.

2. Recommendations: List recommended practices, such as to explain any specific usage that is desired but not required, to explain any entities that are preferred over this one and what entity which application is good for and to itemize exactly what this entity should be used for.

3. Restrictions: List specific restrictions such as forms and options that are not allowed. These are additional restrictions to those in IGES Version 5.1.If None is specified, only the restrictions in IGES apply.

3.4.1 Entity 0: Null Entity

3.4.1.1 Usage

Used to remove an entity from the current file without renumbering the entire file.

3.4.1.2 Recommendations

This entity is a good method for manually removing entities from a specific IGES file without utilizing much of the user's time by not having to reorder and repack an IGES file.

3.4.1.3 Restrictions

None.

3.4.2 Entity 100: Circular Arc

3.4.2.1 Usage

Used to transfer circular arcs, including full circles.

3.4.2.2 Recommendations

A circular arc should be transferred through this entity, although Entity Type 126 could be used. The receiving

system may convert the data to a B-spline format as necessary.

3.4.2.3 Restrictions

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.3 Entity 102: Composite Curve

3.4.3.1 Usage

Used to transfer a curve composed of several parametrized curves. Note that a composite curve entity is not allowed as a component of a composite curve entity.

3.4.3.2 Recommendations

None.

3.4.3.3 Restrictions

A connect point entity (not a NASA-IGES entity) or a point entity in a composite curve should be ignored. This does not invalidate the geometry of a composite curve. However, if a parametric spline curve (not a NASA-IGES entity) is in a composite curve, the composite curve should be ignored by a non-restrictive reader.

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.4 Entity 104: Conic Arc

3.4.4.1 Usage

This entity can be used to represent many types of conic sections.

3.4.4.2 Recommendations

It is recommended that this entity **not** be used. Conics can be accurately represented by B-splines (Entity Type 126). In order to maintain compatibility with many older systems, this entity is included in this specification.

If the sending system knows the conic type, the form of this entity should be set to indicate the type. The entity should be put into its canonical position by the sending system as indicated in Appendix C of IGES V5.1.

3.4.4.3 Restrictions

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.5 Entity 106: Copious Data

3.4.5.1 Usage

Used to transfer an ordered list of points.

3.4.5.2 Recommendations

This entity with Forms 1 to 3 is recommended for transferring a list of points from a cross-section curve. However, the cross-section curve itself should be transferred, instead of points on the curve, since the curve retains more information which may be useful in the receiving system.

In addition to the point coordinate data, a vector is associated with every point in the parameter data section of this entity with Form 3. It is recommended that, if Form 3 is used, this vector be set as the direction vector of the cross section curve. For other recommended usages of this entity, see section 3.4.17.

3.4.5.3 Restrictions

Only Forms 1 to 3 are included in this specification. This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.6 Entity 110: Line

3.4.6.1 Usage

Used to transfer line segments.

3.4.6.2 Recommendations

It is preferred to transfer line segments by this entity than by Entity Type 126, since this is a commonly used and more compact representation.

3.4.6.3 Restrictions

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.7 Entity 116: Point

3.4.7.1 Usage

Used to transfer a point in space.

3.4.7.2 Recommendations

The list of points on a curve or the mesh of points on a surface should be transferred through the appropriate entities, see Entity Type 106 and Entity Type 402.

3.4.7.3 Restrictions

The pointer (PD Index 4) in the parameter data section, which points to the subfigure definition entity specifying the display symbol, will be ignored. The display symbol will be determined by the receiving system.

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.8 Entity 124: Transformation Matrix

3.4.8.1 Usage

Used to transform an entity from its local coordinate system to its true model space position. A number of entities are required by IGES to be transferred in their canonical definition space. For these entities, a transformation matrix is required to relocate them to their true position.

3.4.8.2 Recommendations

None.

3.4.8.3 Restrictions

Only form 0 and form 1 are included in this Specification. The other forms, for view transformation and finite element modeling, are not included.

3.4.9 Entity 126: Rational B-Spline Curve

3.4.9.1 Usage

This format is used as the primary entity for curve transfer. All the other curve types, excluding lines (entity type 110), conics (Entity Type 104) and circular arcs (entity type 100), must be converted (maybe with approximation) to this entity for transfer.

3.4.9.2 Recommendations

This is the most flexible format to represent curves and is recommended for transferring all curves. All lines, circular arcs, and conics can be represented by this entity. This entity contains forms which identify each curve type. If the sending system knows the form of the curve, the form of this entity should be set appropriately.

All parametric splines can also be represented by this entity. Software for the required conversion to this entity can be obtained from NIST (ref. 5).

3.4.9.3 Restrictions

None.

3.4.10 Entity 128: Rational B-Spline Surface

3.4.10.1 Usage

Used as the primary entity for surface transfer. All the other surface types must be converted (maybe with approximation) to this entity for transfer.

3.4.10.2 Recommendations

This is the most flexible format to represent surfaces and is recommended for transferring all surfaces. This entity has forms for some analytic surfaces. If the sending system knows the form of the surface, the Form of this entity should be set appropriately.

3.4.10.3 Restrictions

None.

3.4.11 Entity 141: Boundary

3.4.11.1 Usage

This entity should be used with Entity Type 143. It describes one boundary of a bounded surface.

3.4.11.2 Recommendations

None.

3.4.11.3 Restrictions

There are two Types in this entity. Type 0 transfers only model space curves, and the surface may not be

parametric. Type 1 transfers both parameter and model space curves, and the surface has to be parametric. Only Type 1 is used in this specification.

3.4.12 Entity 142: Curve on a Parametric Surface

3.4.12.1 Usage

This entity is used to transfer a curve on a parametric surface when its parameter space curve is important. A curve on a parametric surface may be a curve from the projection of another curve onto the surface, a curve from the intersection of two surfaces, or an isoparametric curve.

3.4.12.2 Recommendations

None.

3.4.12.3 Restrictions

IGES provides the curve on parametric surface entity for use in either of two ways. It can be used with the trimmed surface entity (Type 144) to form a trimmed surface. Entity 144 is not allowed under this specification so this use is not allowed. The boundary entity (Type 141) should be used for this purpose.

The other use for this entity is to simply represent a curve on a surface. This is the only use allowed for this entity under this specification.

3.4.13 Entity 143: Bounded Surface

3.4.13.1 Usage

Used to transfer a bounded surface, a surface whose domain space is relimited (trimmed back) from its original domain. It should be used with Entity Type 141, Boundary Entity.

3.4.13.2 Recommendations

This entity should be used instead of Entity Type 144, the Trimmed Parametric Surface, for a surface with relimited domain, since Entity Type 144 disallows surfaces with poles or seams, which limits its usage.

3.4.13.3 Restrictions

There are two Types in this entity. Type 0 transfers only model space curves, and the surface may not be parametric. Type 1 transfers both parameter and model space curves, and the surface has to be parametric. Only Type 1 is used.

3.4.14 Entity 212: General Note

3.4.14.1 Usage

Used to pass textual information about the geometry. This can include such information as the history of the object, relevant airfoil section numbers, and reference documents. A general note entity can exist separately or can be associated with another entity or entities.

3.4.14.2 Recommendations

This entity is recommended as the entity for transferring relevant nongeometric design information.

3.4.14.3 Restrictions

Form 0, which states that the text strings in the note are not related to each other positionally, is the only form included in this specification. This is also the default form. Font 1, the default font style for the ASCII character set, is the only font included in this specification. This allows the receiving system to use its default font for display.

3.4.15 Entity 308: Subfigure Definition

3.4.15.1 Usage

The subfigure definition and subfigure instance entities allow one copy of the geometry to be placed in many locations in a design without duplicating the geometry. For example, in a turbine engine design, all the turbine blades on the same stage are identical in shape. Only the geometry for one generic blade must be defined by using a subfigure definition entity. All the blades can then be created with the subfigure instance entity.

3.4.15.2 Recommendations

The user should be discriminatory and exercise sound judgement in using this entity. For example, it is a good practice to represent turbine blades with instances since this reduces file sizes tremendously and makes processing of the files much easier. However, to represent the two wings of an aircraft with an instance may not be wise. Since the geometry for the wings is not stored explicitly in an instance, if the user decides to build a CFD grid on the wings, the grid generation software must create the geometry first. The grid generation software will probably not have the capability to create the geometry for an instance.

3.4.15.3 Restrictions

This entity is not allowed in NASA-IGES-NURBS-Only files.

3.4.16 Entity 314: Color Definition

3.4.16.1 Usage

Used to define additional colors (there are 9 predefined colors in IGES).

3.4.16.2 Recommendations

None.

3.4.16.3 Restrictions

None.

3.4.17 Entity 402: Associativity Instance

3.4.17.1 Usage

This entity is used to group geometry entities into classes. It contains pointers to the grouped entities, called the members of the class. There are 18 predefined forms (classes), of which 4 (Forms 1, 7, 14, 15) are for grouping. Forms 1 and 7 are for unordered groups, i.e., the entities pointed to by this entity are an unordered set. Forms 14 and 15 are for ordered groups, i.e., there is an order specified for the entities pointed to by this entity; the order is defined by the sequence of the pointers specified within this entity.

Unordered groups are frequently used to group surfaces from the same object, hence creating one group per object.

3.4.17.2 Recommendations

Currently, very few CAD systems utilize the ordered group forms. Ordered groups are recommended for grouping a sequence of cross sections and associating them as a surface.

In the recommended usage of the ordered forms, it is not required that the curves be from one surface; this information is irrelevant to this entity. The curves could be sliced from numerous surfaces. In this usage, the members of the class will be the cross section curves.

Ordered groups are also recommended to define a mesh of points (either topologically rectangular or nonrectangular) from a surface. In this usage, the members of the class will be the copious data entity (Entity Type 106, Forms 1-3). The same format is recommended to transfer points on a surface sampled along a list of cross section curves on the surface.

3.4.17.3 Restrictions

Only Forms 1, 7, 14, 15 are included in this specification.

3.4.18 Entity 406: Property, Form 15: Name

3.4.18.1 Usage

This entity is used to associate a name (or brief description) to an entity or a group of entities. All it contains is a text string which is the name.

3.4.18.2 Recommendations

This entity would be appropriate for grouping a portion of the object together, such as a wing, and assigning it a name "wing." Longer comments should be handled through the general note entity.

3.4.18.3 Restrictions

None.

3.4.19 Entity 408: Singular Subfigure Instance

3.4.19.1 Usage

The singular subfigure instance entity creates one instance of a subfigure, which is defined by a subfigure definition entity. See section 3.4.15 for more information.

3.4.19.2 Recommendations

See section 3.4.15.

3.4.19.3 Restrictions

This entity is not allowed in NASA-IGES-NURBS-Only files.

4.0 Validation Methods for NASA-IGES Processors

NASA personnel are developing software for reading, writing, and translating some forms of IGES, NASA-IGES, and NASA-IGES-NURBS-Only files. We are also developing IGES, NASA-IGES, and NASA-IGES-NURBS-Only compliant data files and a test procedure for determining software and data file compliance with this Specification. This information will be contained in a separate document as it becomes available.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	April 1994	Reference Publication

4. TITLE AND SUBTITLE

NASA Geometry Data Exchange Specification for Computational Fluid Dynamics (NASA IGES)

5. FUNDING NUMBERS

505-59

6. AUTHOR(S)

Matthew W. Blake, Patricia A. Kerr*, Scott A. Thorp†, and Jin J. Jou*

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Ames Research Center
Moffett Field, CA 94035-1000

8. PERFORMING ORGANIZATION REPORT NUMBER

A-93089

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

NASA RP-1338

11. SUPPLEMENTARY NOTES

Point of Contact: Matthew Blake, Ames Research Center, MS TO27B-2, Moffett Field, CA 94035-1000;
*Langley Research Center, Hampton, Virginia; †Lewis Research Center, Cleveland, Ohio; *Computer Science Corporation

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unclassified — Unlimited
Subject Category 31

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This document specifies a subset of an existing product data exchange specification that is widely used in industry and government. The existing document is called the Initial Graphics Exchange Specification. This document, a subset of IGES, is intended for engineers analyzing product performance using tools such as computational fluid dynamics (CFD) software. This document specifies how to mathematically define and exchange the geometric model of an object. The geometry is represented utilizing non-uniform rational B-splines (NURBS) curves and surfaces. Only surface models are represented; no solid model representation is included.

This specification does not include most of the other types of product information available in IGES (e.g., no material properties or surface finish properties) and does not provide all the specific file format details of IGES.

The data exchange protocol specified in this document is fully conforming to the American National Standard (ANSI) IGES 5.2.

14. SUBJECT TERMS

Initial Graphics Exchange Specification (IGES) , Computational fluid dynamics, Geometry data exchange

15. NUMBER OF PAGES

52

16. PRICE CODE

A04

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		